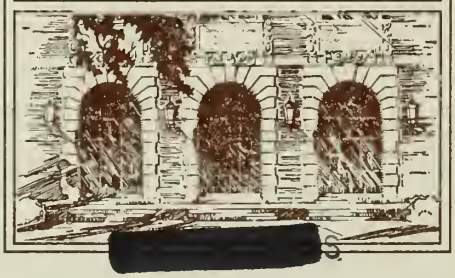
LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84
Il6r
no.776-781
cop.2



The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

**Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.**

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

L161—O-1096

Digitized by the Internet Archive
in 2013

http://archive.org/details/theoreticallimit776shap

510.84
Ilfr
no. 776

UIUCDCS-R-75-776

# THEORETICAL LIMITATIONS ON THE USE OF PARALLEL MEMORIES

by

Henry David Shapiro

December, 1975

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

**Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.**

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

APR 6 1976

L161—O-1096

# THEORETICAL LIMITATIONS ON THE USE OF PARALLEL MEMORIES

Henry David Shapiro, Ph.D.
Department of Computer Science
University of Illinois at Urbana-Champaign, 1976

311

Regardless of the underlying machine architecture, independently addressable memory modules contribute significantly to program speed-ups on modern computers. Because of memory conflicts which arise while accessing data, actual program speed-ups are generally less than theoretically possible. Organizing the data of a computation so as to avoid memory conflicts is particularly difficult for data which can logically be viewed as two-dimensional. Several geometric and algebraic conditions are presented which determine if the data of a computation can be organized to avoid memory conflicts. It is shown that a prime number of memory modules gives higher memory utilization and allows the use of simpler storage schemes than a power of two number of memory modules. The case of greatest practical significance, references to rows, columns and diagonals of a matrix, is given special attention. Finally, a brief discussion is presented which relates this research to that of a companion problem, the construction of memory-processor connection networks for single-instruction-multiple-data stream machines.

UIUCDCS-R-75-776

# THEORETICAL LIMITATIONS ON THE USE OF PARALLEL MEMORIES

by

Henry David Shapiro

B.A., Johns Hopkins University, 1968
M.S., Stanford University, 1969

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois

This work was supported in part by the National Science Foundation under grant no. NSF GJ 41538 and was submitted in partial fulfillment for the Doctor of Philosophy degree in Computer Science, 1975.

510.84
I16r
no.776-781
cop.2



# ACKNOWLEDGMENTS

Special thanks are due to the chairman of my doctoral committee, Professor C. L. Liu, for his constant encouragement and guidance throughout this research. Appreciation is also extended to the other members of the committee, Professors David Kuck, Duncan Lawrie, Judith Liebman, and Franco Preparata, for their many constructive suggestions. Professor Preparata deserves a special note of gratitude for his comments which helped to simplify the proof of Theorem 5.

Also, sincere appreciation is extended to two fellow graduate students, Bruce Link, for his sustained interest in this research as well as his participation in discussions of these results; and Brian Hansche, for our discussions of the conjectures presented in Chapter 4.

To Mr. Stanley Zundo of the Computer Science drafting department goes a note of thanks for his assistance in the preparation of the numerous figures included in this dissertation. Thanks also goes to Mrs. Connie Slovak for an outstanding job in the typing of this manuscript.

Recognition for the financial support which made this research possible is due to both a National Science Foundation Graduate Fellowship and NSF Grant GJ 41538.

Finally, a note of thanks to my wife, Jacqueline, who more than anyone else, encouraged and cheered me when this work progressed slowly.

TABLE OF CONTENTS

Page

1. THE DATA ORGANIZATION PROBLEM..... 1

1.1 Machine Models and the Data Organization Problem..... 1

1.2 Formalization of the Problem..... 7

1.3 Elimination of Boundary Conditions..... 12

1.4 Classes of Skewing Schemes and Some Particular Generalized Lines..... 15

1.5 Summary..... 22

2. DETERMINATION OF VALID SKEWING SCHEMES..... 24

2.1 Introduction..... 24

2.2 The Basic Result..... 24

2.3 Existence and Construction of Valid Skewing Schemes when the Number of Memory Modules Equals the Length of the Generalized Line..... 32

2.4 Existence and Construction of Valid Periodic Skewing Schemes..... 44

3. SPECIAL RESULTS ON $[x,y]_N$-LINES..... 52

3.1 Introduction..... 52

3.2 Preliminaries..... 52

3.3 The Special Case of a Prime Number of Memory Modules..... 56

3.4 Generalization to Composite N..... 60

3.5 Further Results and Examples..... 73

Page

4. UNRESOLVED PROBLEMS AND DIRECTIONS OF FURTHER RESEARCH..... 80

4.1 The Effectiveness of Linear and Periodic Skewing Schemes.... 80

4.2 Questions Relating to Memory Utilization..... 93

4.3 Comments on Broader Problems..... 97

LIST OF REFERENCES..... 100

VITA ..... 102

# LIST OF FIGURES


| Figure | | Page |
|---|---|---|
| 1 | Multi-function Computer..... | 2 |
| 2 | Data Needed for the Evaluation of the Function, F..... | 5 |
| 3 | Parallel Computer..... | 6 |
| 4 | Geometric Realization of a Generalized Line..... | 9 |
| 5 | The Instance of the $[x,y]_N$-line whose First Component is $(i,j)$..... | 19 |
| 6 | Containment Relations Between Classes of Skewing Schemes... | 23 |
| 7 | Instances of a Generalized Line, with their Designated Elements Marked with Asterisks..... | 26 |
| 8 | Checking the Condition in Theorem 4..... | 28 |
| 9 | Pictorial Presentation of the Proof of Theorem 4..... | 30 |
| 10 | Pictorial Presentation of the Proof of Theorem 4..... | 31 |
| 11 | Tesselation of the Plane by a Generalized Line..... | 33 |
| 12 | The Generalized Line $L = ((0,0),(0,1),(0,2),(1,1),(2,1))$ Cannot Tesselate the Plane..... | 34 |
| 13 | The Skewing Scheme Resulting From the Use of Theorem 5..... | 38 |
| 14 | Tesselation of the Plane by the Generalized Line, $L = ((0,0),(1,0),(1,1),(2,0),(2,1))$..... | 41 |
| 15 | Tesselation of the Plane by the Generalized Line, $L = ((0,0),(1,0),(2,0),(2,1),(2,2))$..... | 42 |
| 16 | One Possible Tesselation of the Plane by the Generalized Line, $L = ((0,0),(1,-1),(1,0),(1,1),(2,0))$..... | 43 |
| 17 | Another Possible Tesselation of the Plane by the Generalized Line, $L = ((0,0),(1,-1),(1,0),(1,1),(2,0))$..... | 45 |

| Figure | | Page |
|---|---|---|
| 18 | The "Wrap Around" Interpretation of a Generalized Line Used with Periodic Skewing Schemes..... | 46 |
| 19 | A Valid Linear Skewing Scheme for the Generalized Line, L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2))..... | 50 |
| 20 | Proof of the Existence of a Periodic Skewing Scheme for L = ((0,0),(0,1),(2,1),(2,2))..... | 51 |
| 21 | $[x,y]_N$-lines on the Torus..... | 54 |
| 22 | Programmer's View of STARAN Memory..... | 77 |
| 23 | The Periodic Skewing Scheme Used in the STARAN Computer.... | 78 |
| 24 | Positioning Four Instances of the Generalized Line ((0,0),(1,0),(1,1),(2,0),(2,1)), so their Designated Elements Form a Parallelogram..... | 82 |
| 25 | Alternate Positionings of Instances for the Generalized Line ((0,0),(1,0),(1,1),(1,2),(2,2))..... | 84 |
| 26 | A Non-periodic Skewing Scheme, $\psi$, for which $\varphi(i,j) = \psi(i \bmod N, j \bmod N)$ is not Valid..... | 88 |
| 27 | Examples of Translating by (p,q) and/or (r,s), so that all the Components of the Instance of the Generalized Line Lie Interior to the Parallelogram..... | 90 |
| 28 | Components of an Instance of a Generalized Line, after Translation by (p,q) and/or (r,s), which are Stored in Same Memory Module..... | 91 |
| 29 | An Example of a Polyomino for which There is a Valid Periodic Skewing Scheme, but no Valid Linear Skewing Scheme..... | 92 |
| 30 | Covers for the Generalized Line ((0,0),(0,1),(0,2),(1,1),(2,1)) which Tesselate the Plane..... | 96 |

# 1. THE DATA ORGANIZATION PROBLEM

## 1.1 Machine Models and the Data Organization Problem

In the late 1950's and early 1960's computer architects began to explore the possibility of increasing the speed at which existing computers operated, by performing some internal operations simultaneously. The overlapping of memory fetches with instruction decoding and execution was the basis of the increase in speed of several machines. A difficulty imposed by the hardware technology of that day was that the rate at which the control unit and arithmetic processor could manipulate data was higher than the rate at which a single memory unit could supply the data. Despite many changes in memory and circuit technology over the past twenty years, the inability of a single memory unit to satisfy the data demands of the central processor has not changed. It appears that this situation will persist in the foreseeable future. Because of the relatively slow memory data rate, primary memory on most modern computers consists of several independent memory modules. Since memory fetches can go on simultaneously in different memory modules, the rate at which the memory can supply data to the central processor is effectively increased. A very successful machine, designed on these general principles, was the CDC 6600 [15]. Figure 1 depicts a block diagram of a computer, which may be regarded as an abstraction of this machine. The designers of the CDC 6600 realized that effective use of the potentially highly overlapped functioning of their computer depended on reducing data dependencies in

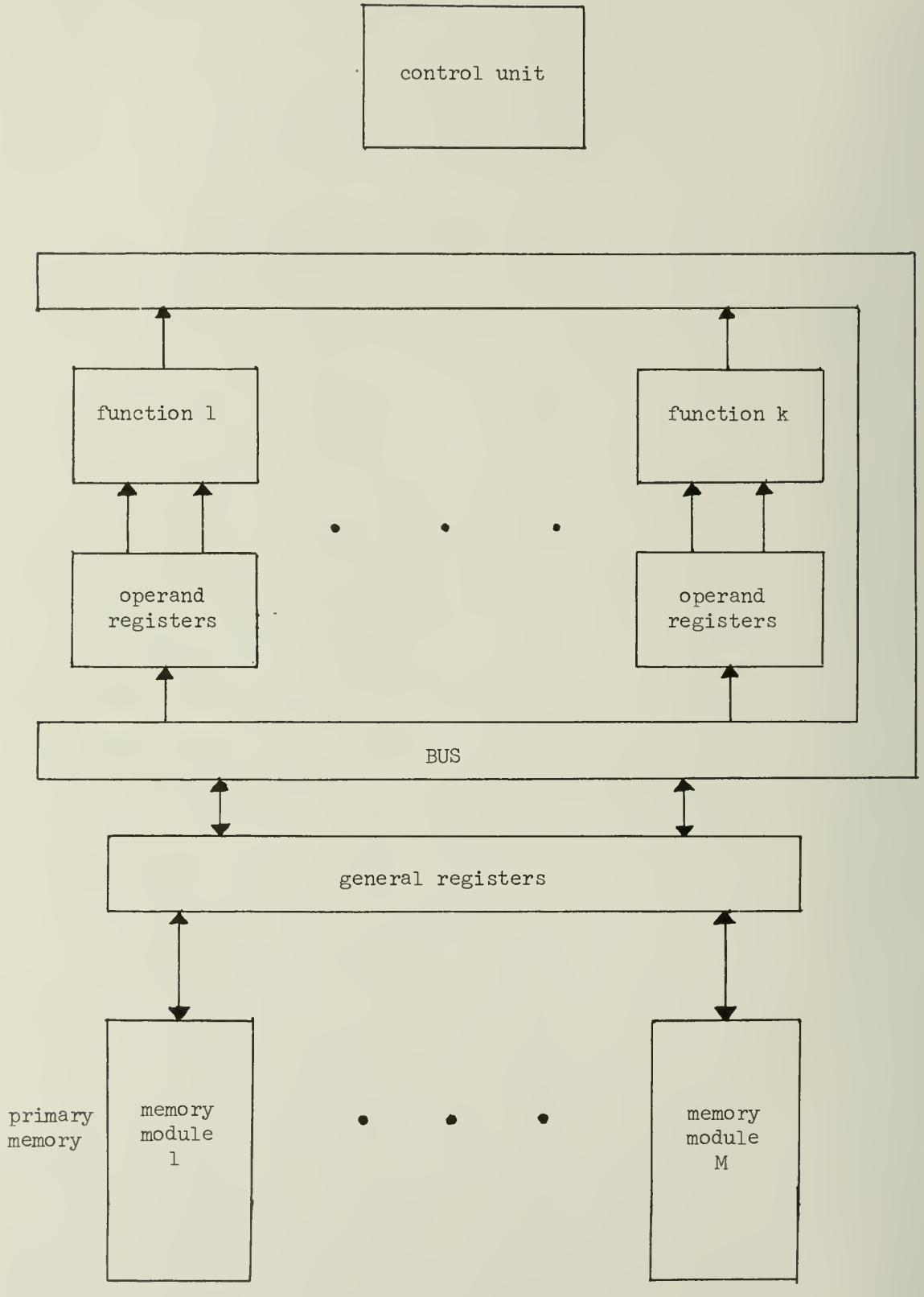


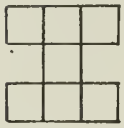

Figure 1: Multi-function Computer.

computations and on storing the data requested by the control unit so that data elements demanded in quick succession were in different memory modules. The problem of detecting and reducing data dependencies in a computation has received much serious attention in the literature. The question of how to organize the data, so that data elements needed in quick succession by the central processor were not in the same memory module, was, for a long time, generally ignored. The designers of the CDC 6600 tried to lessen contention for memory by arranging primary memory so that references to consecutively numbered memory locations cycled through all thirty-two memory modules before repeating. This scheme eliminates memory conflict in the most common cases: Fetching sequential instructions and manipulating the data of one-dimensional arrays stored in consecutive memory locations.

The problem of organizing the data of a two-dimensional array, so that data requested in quick succession are in different memory modules, was left to the programmer and/or the compiler. To make this problem more explicit consider the following specific example. A FORTRAN programmer writes

```
      :
      :
DIMENSION A(N,N)
      :
      :
A(I,J)=F(A(I-1,J-1),A(I-1,J),A(I-1,J+1),A(I,J),A(I+1,J-1),A(I+1,J),A(I+1,J+1))
      :
      :
```

Normally the programmer envisions the memory allocated to array A as actually being two-dimensional, leaving it to the compiler to convert the doubly subscripted references to real machine addresses. Fetching the parameters

for the function call can be thought of as fetching, in rapid succession, the data enclosed by the  in Figure 2. Depending on the dimensions of the array A, and the method of data organization employed by the FORTRAN compiler, some of the seven parameters needed by the function, F, may lie in the same memory module. If such a memory conflict occurs, the fetching of the data, and the overall computation, will be slowed. In general, organizing the data of a two-dimensional array so that memory conflicts are reduced or eliminated is a very difficult problem.

Another machine design in which this same type of problem arises is depicted in Figure 3. This is a single-instruction-multiple-data stream (SIMD) machine, an abstraction of ILLIAC IV. In many computations the goal is to fetch M words of data in parallel and then operate on them simultaneously. If even two of the data words to be fetched are in the same memory, all the processors may have to sit idle while a second memory cycle is initiated. This can affect performance dramatically. Because memory conflicts can seriously degrade performance, in machines of this design, organizing the data of a computation so that memory conflicts are avoided can be very important.

The purpose of this thesis is to develop some mathematical conditions which determine if the data of a two-dimensional array can be stored in a primary memory consisting of independent memory modules, so that during a given computation the data requested by the control unit and/or arithmetic processors can be fetched without memory conflicts. In Chapter 1 preliminaries are considered. Chapter 2 provides a general discussion,

| | | | | |
|---|---|---|---|---|
| $a_{i-3,j-2}$ | $a_{i-3,j-1}$ | $a_{i-3,j}$ | $a_{i-3,j+1}$ | $a_{i-3,j+2}$ |
| $a_{i-2,j-2}$ | $a_{i-2,j-1}$ | $a_{i-2,j}$ | $a_{i-2,j+1}$ | $a_{i-2,j+2}$ |
| $a_{i-1,j-2}$ | $a_{i-1,j-1}$ | $a_{i-1,j}$ | $a_{i-1,j+1}$ | $a_{i-1,j+2}$ |
| $a_{i,j-2}$ | $a_{i,j-1}$ | $a_{i,j}$ | $a_{i,j+1}$ | $a_{i,j+2}$ |
| $a_{i+1,j-2}$ | $a_{i+1,j-1}$ | $a_{i+1,j}$ | $a_{i+1,j+1}$ | $a_{i+1,j+2}$ |
| $a_{i+2,j-2}$ | $a_{i+2,j-1}$ | $a_{i+2,j}$ | $a_{i+2,j+1}$ | $a_{i+2,j+2}$ |
| $a_{i+3,j-2}$ | $a_{i+3,j-1}$ | $a_{i+3,j}$ | $a_{i+3,j+1}$ | $a_{i+3,j+2}$ |

Figure 2: Data Needed for the Evaluation of the Function, F.

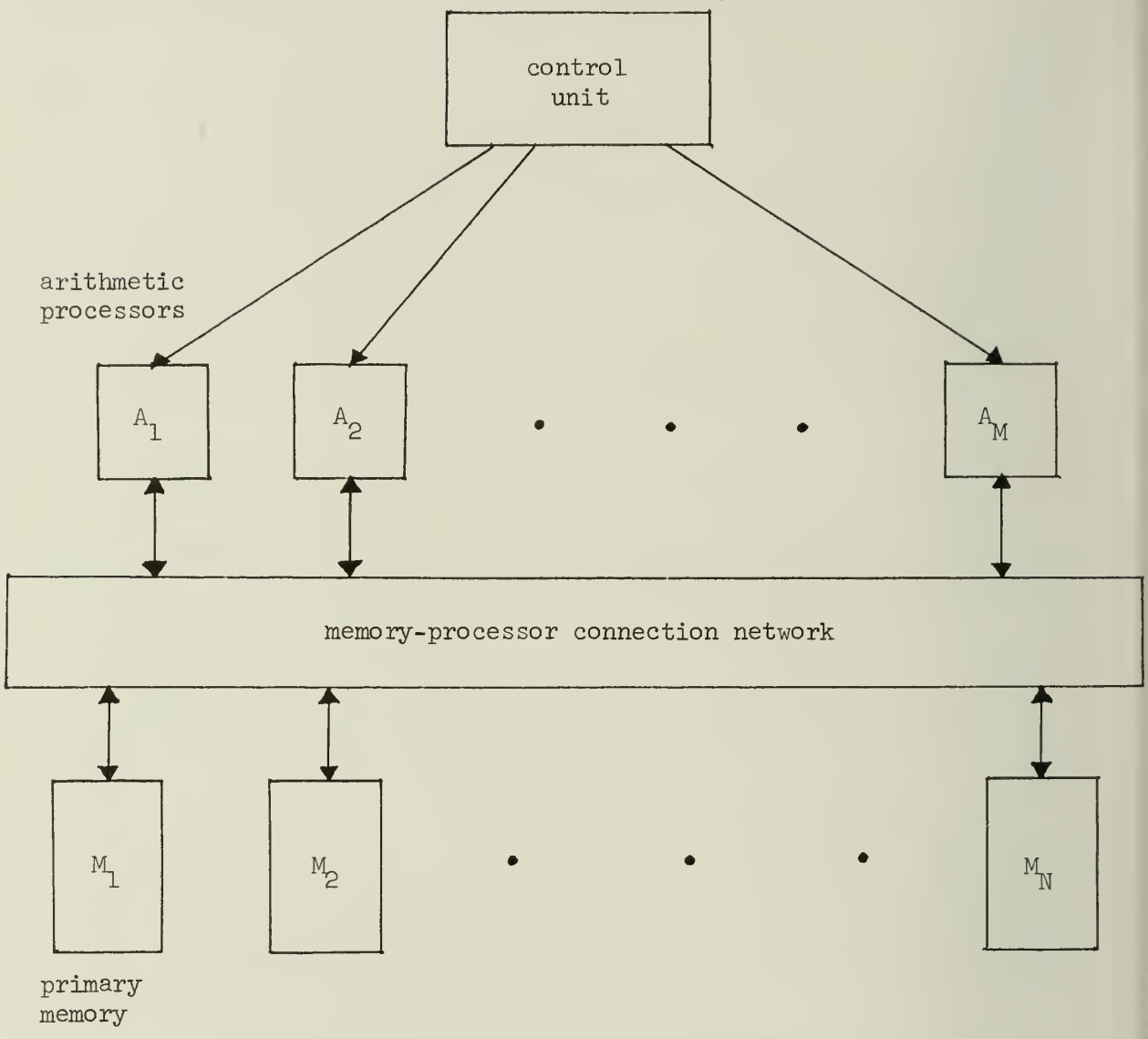


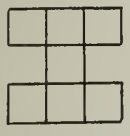
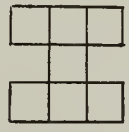

Figure 3: Parallel Computer.

while Chapter 3 focuses attention on some special cases of importance in practice. Chapter 4 informally presents some techniques which have promise in practice, even though complete theoretical analysis of the techniques has not been completed.

## 1.2 Formalization of the Problem

In Section 1.1, the problem of organizing data in parallel memories to eliminate memory conflicts was developed from an historical perspective. To treat this problem mathematically it is convenient to provide a model of the computations that abstract the situation sufficiently so that machine dependent details are eliminated. The data for the computations are to be stored in a doubly subscripted array.

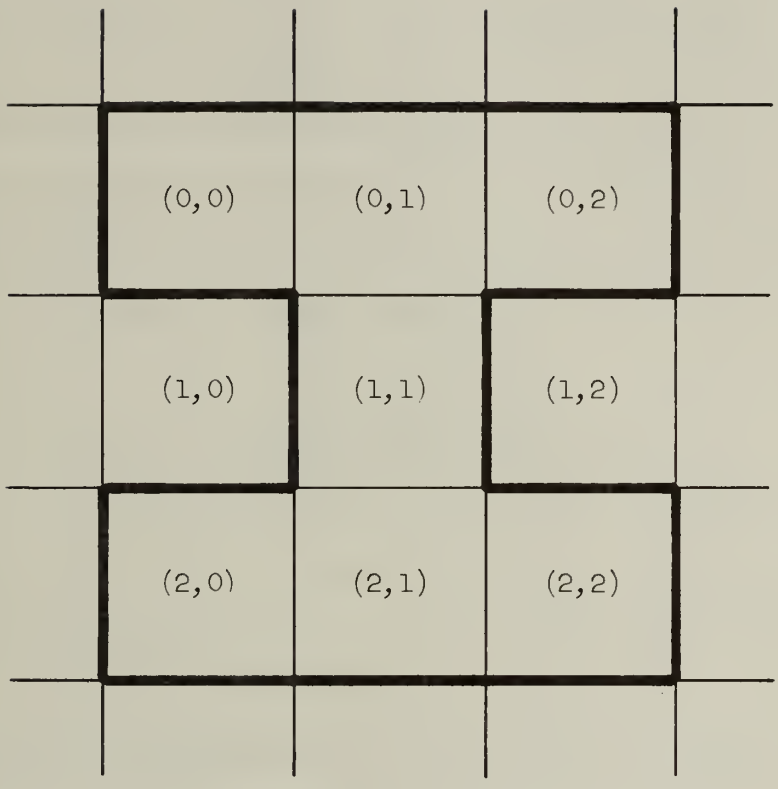
> Definition 1: A generalized line, L, of length n, is an n-tuple of ordered pairs of integers, the first ordered pair of which is (0,0).

A generalized line can be thought of as a rigid template, which during the course of a computation is positioned at various locations over the matrix of data. The data enclosed by the template is to be fetched for a computation. Returning to the programming example used in Section 1.1, the  is to be viewed as the template and its positioning over the matrix of data elements is determined by the actual values of I and J during execution. The data enclosed by the  needs to be fetched before computation of the function, F, can proceed. The actual generalized line is an n-tuple, for example $L_1 = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2))$. This is clearly just a formal way of specifying a

geometric template. The template can be built by placing unit squares on the plane, so that the unit squares are centered at the points of the plane indicated by the ordered pairs of the n-tuple. Figure 4 demonstrates this construction for the generalized line $L_1$.

There are a few minor points that need clarification. First, the labeling of the points of the plane is not the method commonly used in elementary algebra. The first coordinate indicates the vertical direction, with down being positive, and the second coordinate indicates the horizontal direction, with right being positive. This labeling was chosen to reinforce the fact that the data are stored in a two-dimensional array; this method of labeling is often used for two-dimensional arrays in the literature. This labeling scheme also conforms to that of other authors [3,10]. A second minor point is that technically $L_1$ and $L_2$ = ((0,0), (-1,-1),(1,-1),(-1,0),(1,0),(-1,1),(1,1)) are different generalized lines. Their realization by unit squares, however, gives the same geometric shape. A formal definition of equivalent generalized lines could be given; intuitively two generalized lines are equivalent if they realize the same shape, without rotations or reflections. A third point is that the geometric realization of a generalized line need not be a connected figure.

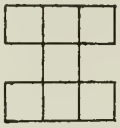
As has been pointed out, a generalized line can be viewed as a template. During the execution of a program this template will be positioned over the matrix of data, and the data elements enclosed by it will be referenced in parallel (or in quick succession, depending on the nature of the machine). The positioning of a template can be viewed as the intuitive interpretation of




Geometric realization of the generalized line, $L_1 = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2))$. The ordered pairs indicate the labeling of points in the plane.

Figure 4: Geometric Realization of a Generalized Line.

Definition 2: An instance of a generalized line, L, is the ordered n-tuple, L(a,b), resulting from the addition of the ordered pair (a,b) to each component of the generalized line.

The positioning of the  in Figure 2 corresponds to $L_2(i,j)$. If the equivalent generalized line, $L_1$, (see Figure 4) is used, then this same collection of data elements corresponds to $L_1(i-1,j-1)$. It is reasonable to consider a version of FORTRAN designed for machines with parallel functioning. The program segment of Section 1.1 might become

```
      ⋮
TEMPLATE L=((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2))
DIMENSION A(N,N)
      ⋮
A(I,J)=F(L(I-1,J-1) OF A)
      ⋮
```

With these definitions it is now possible to give a precise statement of the data organization problem.

Problem: Given a large matrix of data and given that an algorithm requires, at various stages in its computation, the data elements contained in many instances of one or more generalized lines, is it possible to assign the data elements of the matrix to various memory modules, so that when the data elements of an instance of a generalized line are demanded, all the data elements lie in different memory modules?

The ability to store the data, so that when an instance of a generalized line is desired all the data elements are in different memories, is a goal in keeping with the functioning of machines designed along the lines of Figures 1 and 3. If it is possible to store the data so no memory conflicts result when fetching instances of the generalized lines used by the algorithm, then, in machines designed along the lines of Figure 1, contention for memory is lessened, and in machines designed along the lines of Figure 3, the data can be fetched in one memory cycle. This problem motivates the following definitions.

> Definition 3: Given an M×M matrix and N independent memory modules, a skewing scheme is a mapping, $\varphi$: $\{0,1,2,\ldots,M\text{-}1\} \times \{0,1,2,\ldots,M\text{-}1\} \rightarrow \{0,1,2,\ldots,N\text{-}1\}$, where $\varphi(i,j) = k$ means matrix element $a_{i,j}$ is stored in memory module k.

> Definition 4: Given a collection of generalized lines, $\{L_1, L_2, \ldots, L_p\}$, an M×M matrix of data, N memory modules, and a skewing scheme, $\varphi$, the skewing scheme is said to be valid for this collection if and only if given any instance of any of the generalized lines, $\varphi$ assigns the data elements of the instance which lie within the matrix bounds to distinct memory modules.

With these definitions the problem described earlier can be formalized as: Given an M×M matrix, N independent memory modules, and a collection of generalized lines used by an algorithm, is there a valid skewing scheme for this collection?

## 1.3 Elimination of Boundary Conditions

The reader may have noticed in the preceding section that the definition of a skewing scheme explicitly depends on M, the size of the matrix. From the point of view of the programmer this is unfortunate. The matrix size may vary from one run to the next. If a new skewing scheme was needed for each run, use of some programs might be difficult. Notice, however, that if $\varphi$ is a valid skewing scheme for a collection of generalized lines on an $M \times M$ matrix, then $\varphi$ restricted to $\{0,1,2,\ldots,M'-1\} \times \{0,1,2,\ldots,M'-1\}$, $M' < M$, is also valid for this collection on an $M' \times M'$ matrix. The pragmatic consideration that M may not be known in advance, and can be large, leads to the search for valid skewing schemes with domain $\{0,1,2,\ldots\} \times \{0,1,2,\ldots\}$. When skewing schemes with this domain are used several benefits accrue. As noted above, such a skewing scheme can be used without prior knowledge about the size of M. A secondary benefit is that special case handling of instances which overlap the boundaries along the right and bottom of the $M \times M$ matrix can be simplified. Zero or some other null value can be stored for the value of data elements outside the actual array bounds. These practical considerations justify elimination of the boundaries along the right and bottom edges of the matrix, i.e. there are reasons to treat the matrix as infinite in size. It is also possible to show mathematically that widening the domain of skewing schemes to the quarter plane does not result in any loss of generality, that is, those collections for which valid skewing schemes exist on all finite domains, have valid skewing schemes on the quarter plane.

Theorem 1: Given a collection of generalized lines, $\{L_1, L_2, \ldots, L_p\}$, and N, the number of memory modules,

if for each M there exists a valid skewing scheme for this collection, $\varphi_M$: $\{0,1,2,\ldots,M-1\} \times \{0,1,2,\ldots,M-1\} \rightarrow \{0,1,2,\ldots,N-1\}$, then there exists a valid skewing scheme for this collection with domain $\{0,1,2,\ldots\} \times \{0,1,2,\ldots\}$.

Proof: The proof uses the König infinity lemma: If a rooted tree has infinitely many nodes, but each node has finitely many successors, then there is a path of infinite length in the tree [9]. We construct a rooted tree as follows. The nodes at level i in the tree are the skewing schemes valid for this collection for an $i \times i$ matrix. Recall that for instances which overlap the boundaries of the matrix there must not be any memory conflicts for elements of the instance lying inside the matrix bounds. Also note that the one node at level 0, the root, is an artificial construct, since matrices of dimension zero have no data elements, i.e. one node at level 0 is created, by convention, to provide a root for the tree. A node at level i, $\varphi_i$, is connected to a node at level i+1, $\varphi_{i+1}$, if $\varphi_{i+1}$ restricted to $\{0,1,2,\ldots,i-1\} \times \{0,1,2,\ldots,i-1\}$ is just $\varphi_i$. (The node at level 0 is connected to all nodes at level 1, by convention.) This construction produces a tree. To see this, note that every node at level i, $i > 0$, has a predecessor, for if $\varphi_i$ is a valid skewing scheme on the $i \times i$ matrix, then $\varphi_i$ restricted to $\{0,1,2,\ldots,i-2\} \times \{0,1,2,\ldots,i-2\}$ is a valid skewing scheme on the $(i-1) \times (i-1)$ matrix. Also note that each $\varphi_i$ has only one predecessor. Thus the construction yields a tree. Next, notice that the tree has infinitely many nodes, since by assumption there is a valid skewing scheme for every M, and, hence, at least one node at each level. Lastly, each node has only finitely many successors, in fact a

node at level i has exactly $N^{2i+1}$ possible candidates for successor nodes, many of which will presumably fail to be valid skewing schemes. Thus the König infinity lemma implies an infinite path in the tree. Let the nodes in this path be $\psi_0,\psi_1,\psi_2,\ldots$. Define $\varphi$ by $\varphi(i,j) = \psi_k(i,j)$ where $k > \max(i,j)$. Notice that $\varphi$ is a well-defined function, since if $k_2 > k_1$ then $\psi_{k_2}$ restricted to $\{0,1,2,\ldots,k_1\text{-}1\} \times \{0,1,2,\ldots,k_1\text{-}1\}$ is just $\psi_{k_1}$, by the way in which the tree was constructed. To see that $\varphi$ is valid for the collection $\{L_1,L_2,\ldots,L_p\}$, consider an arbitrary instance of one of the generalized lines. Let k be selected sufficiently large so that this instance does not overlap the right or bottom boundaries of the $k \times k$ matrix. Now since $\psi_k$ is valid for this collection on the $k \times k$ matrix, and $\varphi$ restricted to $\{0,1,2,\ldots,k\text{-}1\} \times \{0,1,2,\ldots,k\text{-}1\}$ equals $\psi_k$, all the data elements comprising the instance (except those which overlap the left and top boundaries) will be mapped to different memories by $\psi_k$, and hence $\varphi$. Since the instance was arbitrary, $\varphi$ is a valid skewing scheme. ■

Theorem 1 shows that in searching for a valid skewing scheme, M can be ignored. As pointed out earlier, an additional benefit is that special handling at some matrix boundaries is eliminated. To extend this simplification to the left and top boundaries it is convenient to use $\{\ldots,-1,0,1,\ldots\} \times \{\ldots,-1,0,1,\ldots\}$ as the domain for skewing schemes. Unlike our situation earlier, this change cannot be justified on the practical grounds that the size of the matrix may not be known in advance. However, any difficulties at the left and top boundaries can be eliminated by

Theorem 2: Given a collection of generalized lines, $\{L_1,L_2,\ldots,L_p\}$, and N, the number of memory modules,

if there exists a skewing scheme, $\varphi$, valid for this collection with domain $\{0,1,2,\ldots\} \times \{0,1,2,\ldots\}$, then there is a skewing scheme, $\hat{\varphi}$, valid for this collection with domain $\{\ldots,-1,0,1,\ldots\} \times \{\ldots,-1,0,1,\ldots\}$.

Proof: The proof is similar to the preceding proof, so only a few details are sketched. The main difference is that nodes at level i are chosen to represent skewing schemes valid for the collection of generalized lines with domain $\{-i,-i+1,\ldots,i-1,i\} \times \{-i,-i+1,\ldots,i-1,i\}$. The only new difficulty is to see that level i is not empty. This is so since $\varphi$ restricted to $\{0,1,2,\ldots,2i\} \times \{0,1,2,\ldots,2i\}$ is valid, so $\psi_i$ defined on $\{-i,-i+1,\ldots,i-1,i\} \times \{-i,-i+1,\ldots,i-1,i\}$ by $\psi_i(j,k) = \varphi(j+i,k+i)$ is also valid. ■

The content of Theorem 2 is that there is no loss of generality in considering only matrices of data that are infinite in all directions. This is particularly useful in formulating theoretical results, since proofs no longer need to account for any special conditions that might arise when only part of an instance lies inside the matrix bounds. Because of Theorem 2 only skewing schemes whose domain is the entire plane will be considered throughout the rest of this thesis.

## 1.4 Classes of Skewing Schemes and Some Particular Generalized Lines

Given a collection of generalized lines it is desirable to have some conditions which determine whether or not a valid skewing scheme exists. In situations that arise in actual practice the existence of such a valid skewing scheme is usually not sufficient. It is also highly desirable that $\varphi(i,j)$ be readily calculable, so that address computation does not overly degrade system performance. There are two approaches that can be taken.

One is that $\varphi$ be represented by a simple mathematical formula. The second is to use some table look-up strategy.

Neither of these two methods of calculating $\varphi$ works well for arbitrary skewing schemes. In general, a closed form mathematical expression for an arbitrary skewing scheme, $\varphi$, may not exist. Table look-up techniques will also not be of much help, since a large (theoretically infinite) table will have to be stored, and storing the table in memory so that memory conflicts are eliminated in obtaining information from the table is the same problem as storing the original matrix of data so memory conflicts are eliminated. Because arbitrary skewing schemes cannot always be implemented conveniently, certain subclasses of the skewing schemes valid for the entire plane take on significance. Table look-up schemes motivate the following definition.

> Definition 5: Given N, the number of memory modules, a skewing scheme, $\varphi$, is called <u>periodic</u> if and only if $\varphi(i,j) = \varphi(i+kN, j+\ell N)$, for $k, \ell = \ldots,-2,-1,0,1,2,\ldots$ and for any i and j.

If $\varphi$ is a periodic skewing scheme, then $\varphi(i,j) = \varphi(i \bmod N, j \bmod N)$ Therefore to calculate the value of $\varphi$ at any point in the plane it is only necessary to know $\varphi$ on $\{0,1,\ldots,N-1\} \times \{0,1,2,\ldots,N-1\}$. If N is sufficiently small, periodic skewing schemes can be implemented by table look-up at reasonable cost. The needed values of $\varphi$ can be stored in a specially designed super-fast memory.

As N becomes large, and especially in machines designed along the lines suggested by Figure 3, where each arithmetic unit may require a private copy of the basic $N \times N$ storage map, periodic skewing schemes

realizable only by table look-up become unattractive. In such situations the first method suggested for computing $\varphi$, a simple mathematical formula, appears more reasonable. One class of skewing schemes that has attracted attention in the literature is the class of linear skewing schemes.

Definition 6: Given N, the number of memory modules, a skewing scheme, $\varphi$, is called linear if and only if there exist constants a and b such that $\varphi(i,j) = ai+bj \bmod N$.

The class of linear skewing schemes is a subclass of the periodic skewing schemes, since if $\varphi$ is a linear skewing scheme, then $\varphi(i+kN, j+\ell N) = a(i+kN)+b(j+\ell N) \bmod N = ai+bj \bmod N = \varphi(i,j)$, and, thus, $\varphi$ is a periodic skewing scheme. That the linear skewing schemes are a subclass of the periodic skewing schemes shows that some periodic skewing schemes can be implemented without table look-up. There are other periodic skewing schemes which can also be efficiently implemented without table look-up. The one used in the STARAN computer will be mentioned in Chapter 3.

Budnik and Kuck [3] and Lawrie [10] have investigated linear skewing schemes in detail. Much of their work was motivated by considering machines designed as in Figure 3. After investigating the data requirements of programs written for similar machines, and after discussions with numerical analysts, they generally focused their attention on some commonly used generalized lines. In particular the generalized lines consisting of N consecutive elements of a matrix row (the generalized line $R=((0,0),(0,1),(0,2),\ldots,(0,N-1))$ ), N consecutive elements of a matrix column (the generalized line $C=((0,0),(1,0),(2,0),\ldots,(N-1,0))$ ), N consecutive elements of a forward diagonal (the generalized line

$D=((0,0),(1,1),(2,2),\ldots,(N-1,N-1))$ ), N consecutive elements of a backwards diagonal (the generalized line $B=((0,0),(1,-1),(2,-2),\ldots,(N-1,-N+1))$ ), and, when N was a perfect square, $\sqrt{N} \times \sqrt{N}$ blocks (the generalized line $S=((0,0),(0,1),\ldots,(0,\sqrt{N}-1),(1,0),(1,1),\ldots,(1,\sqrt{N}-1),\ldots,(\sqrt{N}-1,0),(\sqrt{N}-1,1),\ldots,(\sqrt{N}-1,\sqrt{N}-1))$ ) were of primary concern. One of the main results of Budnik and Kuck [3] is that if $2|N$ or $3|N$, where N is the number of memories, then there is no valid linear skewing scheme for the collection of generalized lines, {R,C,D,B}. In order to generalize this result and to provide a reasonable notation, a definition is useful.

Definition 7: An $[x,y]_N$-line is a generalized line of the form $((0,0),(y,x),(2y,2x),\ldots,((N-1)y,(N-1)x))$.†

Pictorially, the template for an $[x,y]_N$-line is formed by starting at the origin and going over x and down y, until a total of N points are generated (see Figure 5). In this notation the generalized line representing N consecutive elements of a row is the $[1,0]_N$-line, the generalized line representing N consecutive elements of a column is the $[0,1]_N$-line, etc. The following result can be found in Budnik and Kuck [3], though different notation is used.

Theorem 3: Given N memory modules and a collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, $\varphi(c,d) = ac + bd \bmod N$ is a valid linear skewing scheme for this collection if and only if $(ay_i+bx_i,N)^{\ddagger} = 1$, for $i=1,2,\ldots,I$.

---

†Note that N is the length of the $[x,y]_N$-line.

‡$(c,d)$ is the greatest common divisor of c and d.

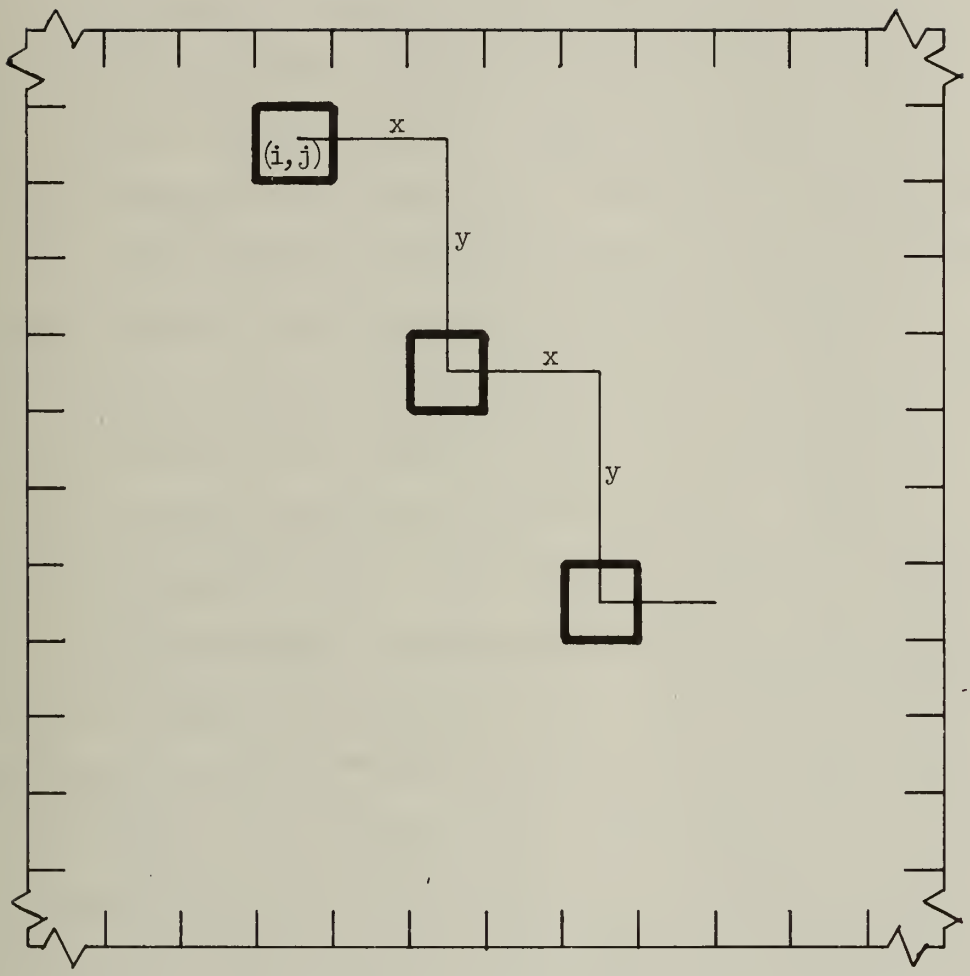



Figure 5: The Instance of the $[x,y]_N$-line whose First Component is (i,j).

Proof: Suppose first that $\varphi(i,j) = ai+bj \bmod N$ is a valid skewing scheme for this collection. Take an arbitrary $[x,y]_N$-line from the collection, say $L = [x_r,y_r]_N$-line. Since $\varphi$ is valid the N elements in the instance L(0,0) must be mapped by $\varphi$ to different memory modules. Since this instance is just $((0,0),(y_r,x_r),\ldots,((N-1)y_r,(N-1)x_r))$, it must be the case that $\varphi(0,0) \neq \varphi(\nu y_r,\nu x_r)$, for $\nu=1,2,\ldots,N-1$. Thus $a\cdot 0+b\cdot 0 \bmod N \neq a\nu y_r+b\nu x_r \bmod N$, for $\nu=1,2,\ldots,N-1$. But this is just $\nu(ay_r+bx_r) \not\equiv_N 0$, for $\nu=1,2,\ldots,N-1$, and it is a well-known result of elementary number theory that this implies $(ay_r+bx_r,N) = 1$ [7].

Conversely suppose there exists a and b such that $(ay_r+bx_r,N) = 1$, for $i=1,2,\ldots,I$. To show that $\varphi(i,j) = ai+bj \bmod N$ is valid for the collection consider an arbitrary instance of an arbitrary $[x,y]_N$-line in the collection. It suffices to show that the N elements in this instance are mapped to different memory modules. For definiteness, consider $L = [x_r,y_r]_N$-line and the instance L(i,j). Since the choice of line and the choice of instance were made arbitrarily, if $\varphi$ maps all the components of $((i,j),(i+y_r,j+x_r),\ldots,(i+(N-1)y_r,j+(N-1)x_r))$ to different memory modules, then $\varphi$ will be valid. But if $\varphi(i+\nu y_r,j+\nu x_r) = \varphi(i+\nu' y_r,j+\nu' x_r)$, for some $\nu,\nu' \in \{0,1,2,\ldots,N-1\}$, with $\nu \neq \nu'$, then $ai+a\nu y_r+bj+b\nu x_r \equiv_N ai+a\nu' y_r+bj+b\nu' x_r$ which implies $(\nu-\nu')(ay_r+bx_r) \equiv_N 0$, and since $\nu-\nu' \not\equiv_N 0$, this contradicts the assumption that $(ay_i+bx_i,N) = 1$ for $i=1,2,\ldots,I$. ■

The result of Budnik and Kuck mentioned earlier follows immediately, since the collection of generalized lines they refer to is just $\{[1,0]_N$-line,$[0,1]_N$-line,$[1,1]_N$-line,$[1,-1]_N$-line$\}$, and one of a, b, and a+b will be divisible by 2, so if $2|N$ no choices of a and b satisfy the conditions of the theorem. Similarly one of a, b, a+b, and -a+b will be divisible by 3.

Lawrie [10] points out that in computers designed along the lines of Figure 3 solving the data organization problem is insufficient in practice. To be able to use such a computer in a reasonable way, an efficient memory-processor connection network, which can route the data to the appropriate arithmetic unit, is also needed. In providing a data organization scheme and a connection network, Lawrie uses $P = 2^{2q}$ processors and $N = 2P$ memories. Within this framework he found that by use of linear skewing he could fetch any P consecutive elements of any row, column, forward diagonal, backward diagonal, or any $\sqrt{P} \times \sqrt{P}$ block without memory conflicts. (Take $a = \sqrt{P} + 1$ and $b = 2$.) This does not violate Theorem 3, since the number of memories, N, is not equal to the length of the lines, P. In addition to providing a skewing scheme, Lawrie also designed a network, the $\Omega$-network, which routed the data to the appropriate processor in $O(\log_2 P)$ time. F. Yao [16] has shown that the $\Omega$-network is optimal. Lawrie left unanswered the question of using some non-linear skewing scheme to achieve the same conflict free access while at the same time reducing the number of memories, N, to be equal to the number of processors, $P = 2^{2q}$. The restriction that the number of processors be a power of two was kept so that any arithmetic mod N could be performed rapidly by shifting and by the hope that the $\Omega$-network, or some slight modification of it would still be able to align the data. In Chapter 3 of this thesis it will be shown that if the number of memories equals the number of processors, which in turn is a power of two, then no skewing scheme of any type will be valid for the collection of generalized lines Lawrie considers.

Swanson [14] has also studied the problem of designing efficient memory-processor connection networks. In his construction the number of memories, N, equals the number of processors, P, and P is prime. For this case he has designed a network based on k-apart shifters which operates in $O(\sqrt{P})$ time, but uses very little hardware. Unfortunately, to align the data with the processors his network must be followed by a shift network, which requires an additional $O(\log_2 P)$ time and $O(\log_2 P)$ hardware. The choice of P as a prime, however, guarantees that any instance of the $[1,0]_P$-, $[0,1]_P$-, $[1,1]_P$-, and $[1,-1]_P$-lines can be fetched without conflict. The question of designing memory-processor connection networks will be discussed again briefly in Chapter 4.

## 1.5 Summary

In this chapter several computer designs which utilize parallel memory modules to achieve program speed-ups were considered. The problem of organizing the data so that computations can proceed efficiently was formalized. Generalized lines, and as special cases, $[x,y]_N$-lines, were defined. Various classes of skewing schemes were also defined. Figure 6 depicts the containment relationships between these classes. Theorem 2 shows that even though skewing schemes valid on the entire plane are a subclass of those valid on the quarter plane, the two classes of skewing schemes are of equal power, in that the collections of generalized lines they can handle are exactly the same. In Chapters 3 and 4 similar results will be presented for linear and periodic skewing schemes, but only for certain subclasses of generalized lines.

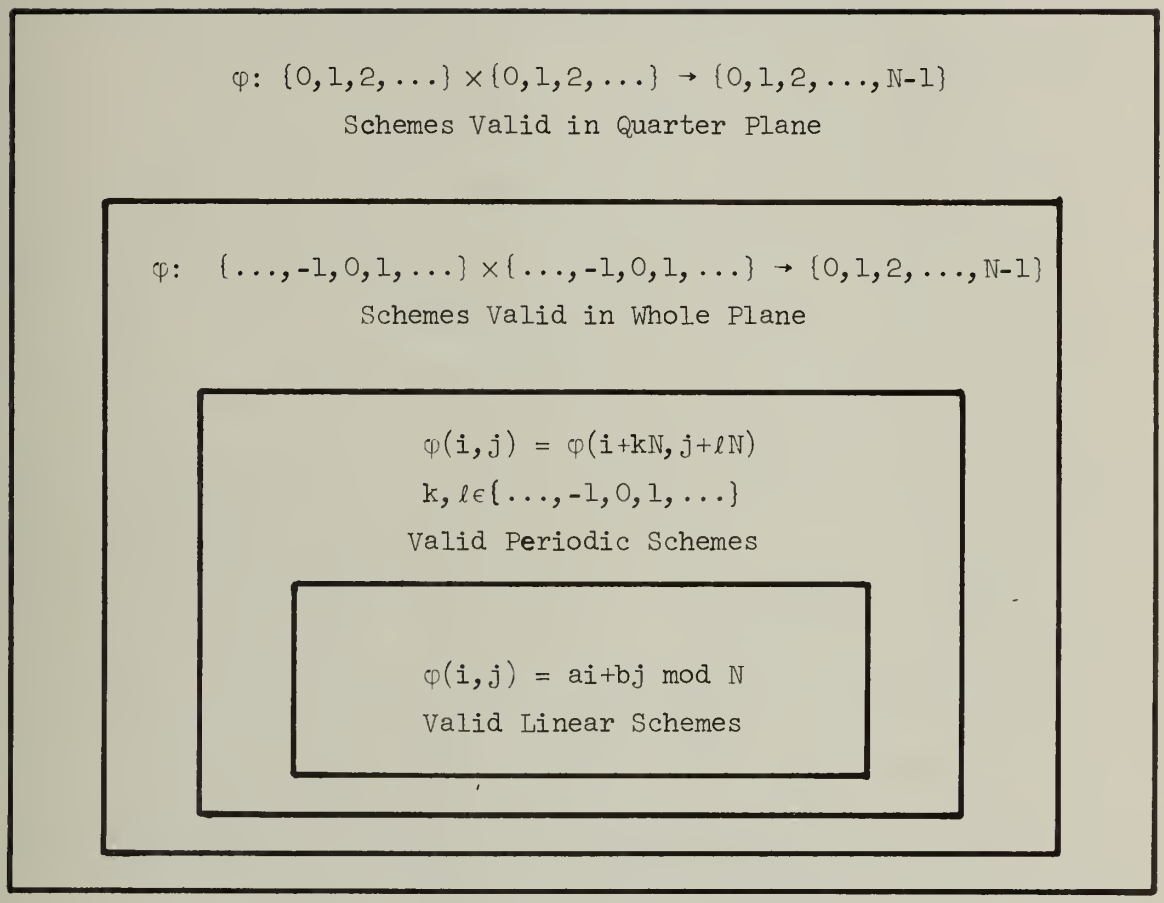



Figure 6: Containment Relations Between Classes of Skewing Schemes.

# 2. DETERMINATION OF VALID SKEWING SCHEMES

## 2.1 Introduction

Chapter 1 dealt primarily with definitions and historical perspectives. In this chapter results will be presented on whether a matrix of data can be stored in N memory modules so that instances of a given generalized line, or collection of generalized lines, can be fetched without memory conflicts. In some of the results the length of the generalized line(s) will be restricted to equal the number of memory modules. This restriction is motivated by the desire to maximally utilize memory in computers designed as indicated in Figure 3. If such a computer has M arithmetic units, then the data elements of instances of generalized lines of up to length M can be processed in parallel. If the number of memories is N, then memory utilization is limited to $\frac{M}{N}$, because only M out of the N memory modules will be referenced in one memory cycle. If 100% memory utilization is desired, N = M is required. Since the memory-processor connection network will be more complex for larger N, requiring N = M has additional benefits.

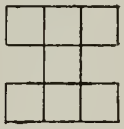
In the last part of the chapter special attention is paid to periodic skewing schemes. The results developed in Sections 2.2 and 2.3 will be seen to carry over to the periodic case without significant change.

## 2.2 The Basic Result

In this section a necessary and sufficient condition is presented for determining the validity of a skewing scheme for a generalized line of

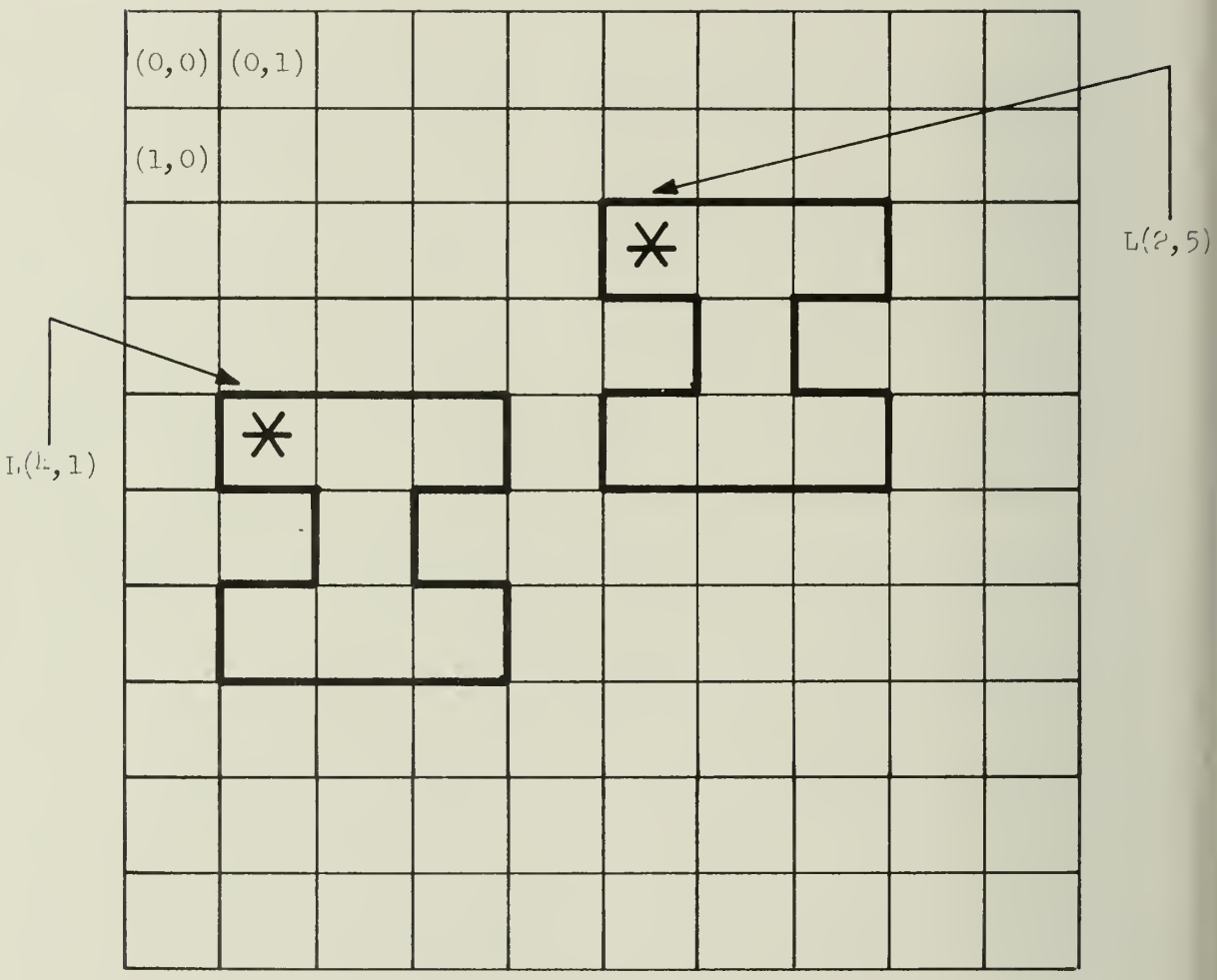
length M, using N memory modules. To do this it is convenient to augment some earlier definitions.

Definition 8: The designated element of a generalized line or an instance of a generalized line is the first ordered pair of the n-tuple.

In the material to follow, the intuitive, geometric viewpoint is a valuable aid to understanding, and, for this reason, the intuitive, as well as the formal, approach will be presented. As discussed in Chapter 1, a generalized line can be viewed as a rigid template. The designated element can then be indicated by marking the appropriate square of the generalized line or an instance of the generalized line by a distinguishing mark, like an asterisk. For the  -shaped generalized line, given by L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2)), this is illustrated in Figure 7.

Theorem 4: Given a generalized line of length M, N memory modules, and a skewing scheme, the skewing scheme is valid for this generalized line if and only if the following condition holds for every $k \in \{0,1,2,\ldots,N-1\}$: When all instances of the generalized line are considered, in which the designated element of the instance is mapped by the skewing scheme into memory k, no two of these instances have an element in common.

Before presenting the formal proof, an intuitive explanation of the condition presented above might be useful. Imagine a two-dimensional memory storage map laid out on the plane, in which the entry at (i,j) is




L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2))

Figure 7: Instances of a Generalized Line, with their Designated Elements Marked with Asterisks.

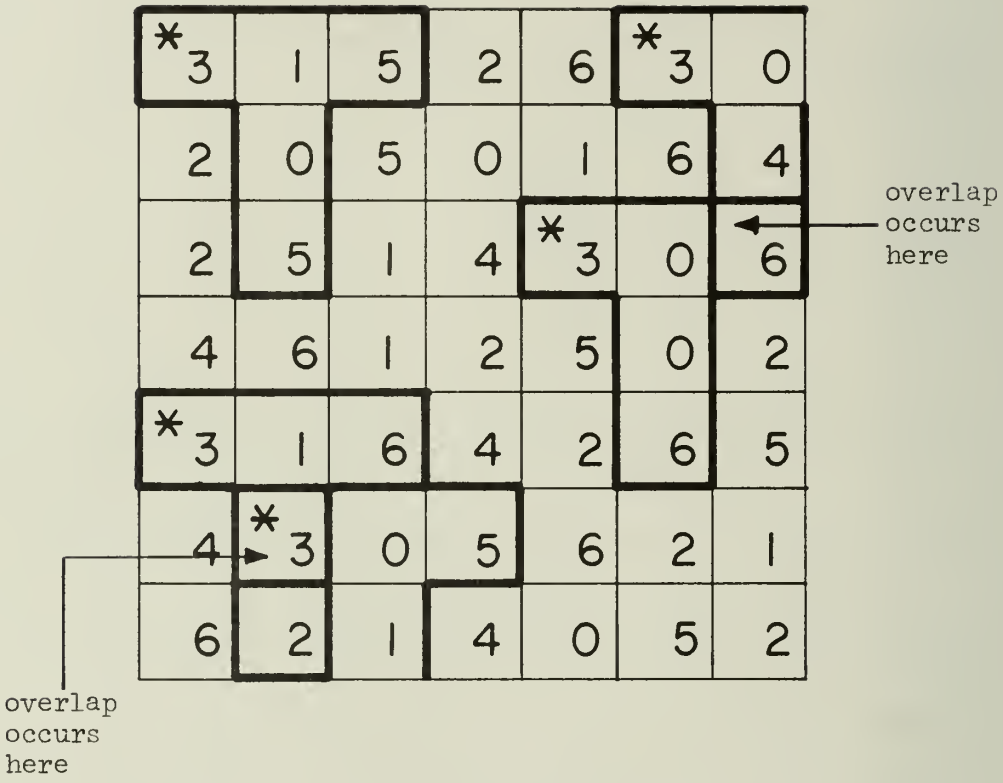
the value of the skewing scheme at this point. Also assume an infinite supply of realizations of the generalized line by unit squares, with the designated element marked with an asterisk. Select a $k \in \{0,1,2,\ldots,N-1\}$. For each location in the memory storage map containing a k, place a copy of the generalized on the plane so that the asterisk is directly over the k. If, after completing this construction, no two instances of the generalized line laid down on the plane, overlap, and this is true irrespective of the choice of k, then the skewing scheme is valid. Conversely, if the skewing scheme is valid, then when the construction described above is performed for any k, the instances will not overlap. This construction is illustrated in Figure 8.

Proof of Theorem 4: Suppose first that for some $k \in \{0,1,2,\ldots,N-1\}$ there are two distinct instances of the generalized line, whose designated elements are mapped by the skewing scheme, $\varphi$, into memory module k, and that these two instances have an element in common. Being specific, let the generalized line be $L = ((x_1,y_1),(x_2,y_2),\ldots,(x_M,y_M))^{\dagger}$ and let the two instances be $L(a_1,b_1) = ((a_1+x_1,b_1+y_1),(a_1+x_2,b_1+y_2),\ldots,(a_1+x_M,b_1+y_M))$ and $L(a_2,b_2) = ((a_2+x_1,b_2+y_1),(a_2+x_2,b_2+y_2),\ldots,(a_2+x_M,b_2+y_M))$. Then, the two instances have an element in common means $(a_1+x_i,b_1+y_i) = (a_2+x_j,b_2+y_j)$, for some i and j. In addition, $i \neq j$, for $i = j$ implies $(a_1,b_1) = (a_2,b_2)$, contrary to the assumption that the instances are distinct. Since the designated elements of these two instances are mapped by $\varphi$ into memory k, $\varphi(a_1+x_1,b_1+y_1) = \varphi(a_2+x_1,b_2+y_1) = k$.

In order to see that $\varphi$ is not a valid skewing scheme for L, consider the instance $L(a_1+x_1-x_j,b_1+y_1-y_j) = ((a_1+x_1-x_j+x_1,b_1+y_1-y_j+y_1),$

---

$^{\dagger}$From the definition of generalized line, $(x_1,y_1) = (0,0)$.



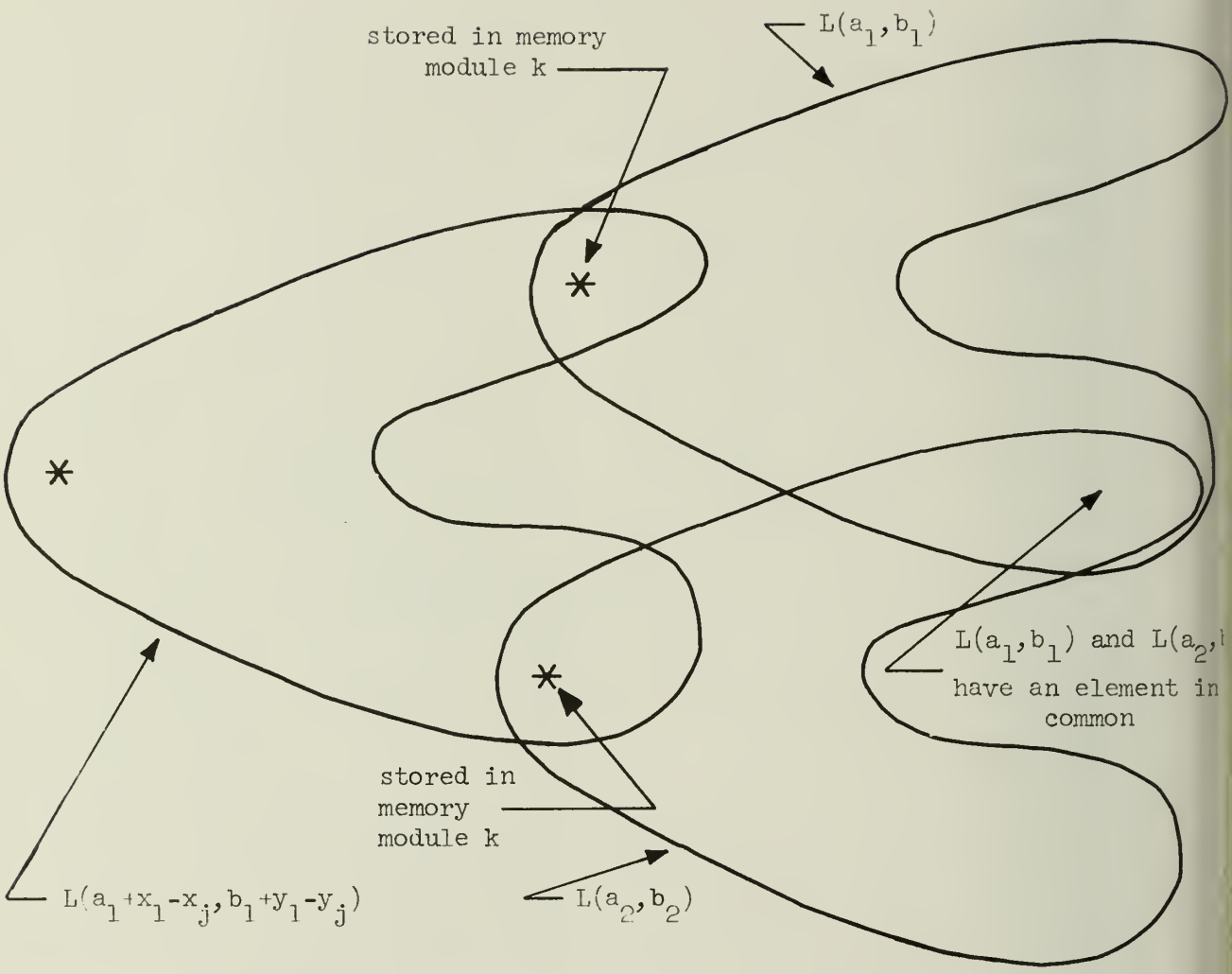

L = ((0,0),(0,1),(0,2),(1,1),(2,1)). N = 7. The condition in Theorem 4 is tested for k = 3. Overlap occurs in two places, implying the skewing scheme is not valid. Only a small section of the plane is shown.

Figure 8: Checking the Condition in Theorem 4.

$(a_1+x_1-x_j+x_2, b_1+y_1-y_j+y_2)$, ..., $(a_1+x_1-x_j+x_M, b_1+y_1-y_j+y_M))$. The $j^{th}$ component of $L(a_1+x_1-x_j, b_1+y_1-y_j)$ is $(a_1+x_1-x_j+x_j, b_1+y_1-y_j+y_j) = (a_1+x_1, b_1+y_1)$, which is mapped by $\varphi$ into memory k. The $i^{th}$ component of $L(a_1+x_1-x_j, b_1+y_1-y_j)$ is $(a_1+x_1-x_j+x_i, b_1+y_1-y_j+y_i) = (a_2+x_j+x_1-x_j, b_2+y_j+y_1-y_j) = (a_2+x_1, b_2+y_1)$, where $(a_1+x_i, b_1+y_i) = (a_2+x_j, b_2+y_j)$ was used. Since $\varphi(a_2+x_1, b_2+y_1) = k$, the $i^{th}$ component of $L(a_1+x_1-x_j, b_1+y_1-y_j)$ is also mapped into memory k. Because two distinct components of this instance are mapped by $\varphi$ into memory k, $\varphi$ is not valid for L. Figure 9 provides an intuitive picture for this part of the proof.

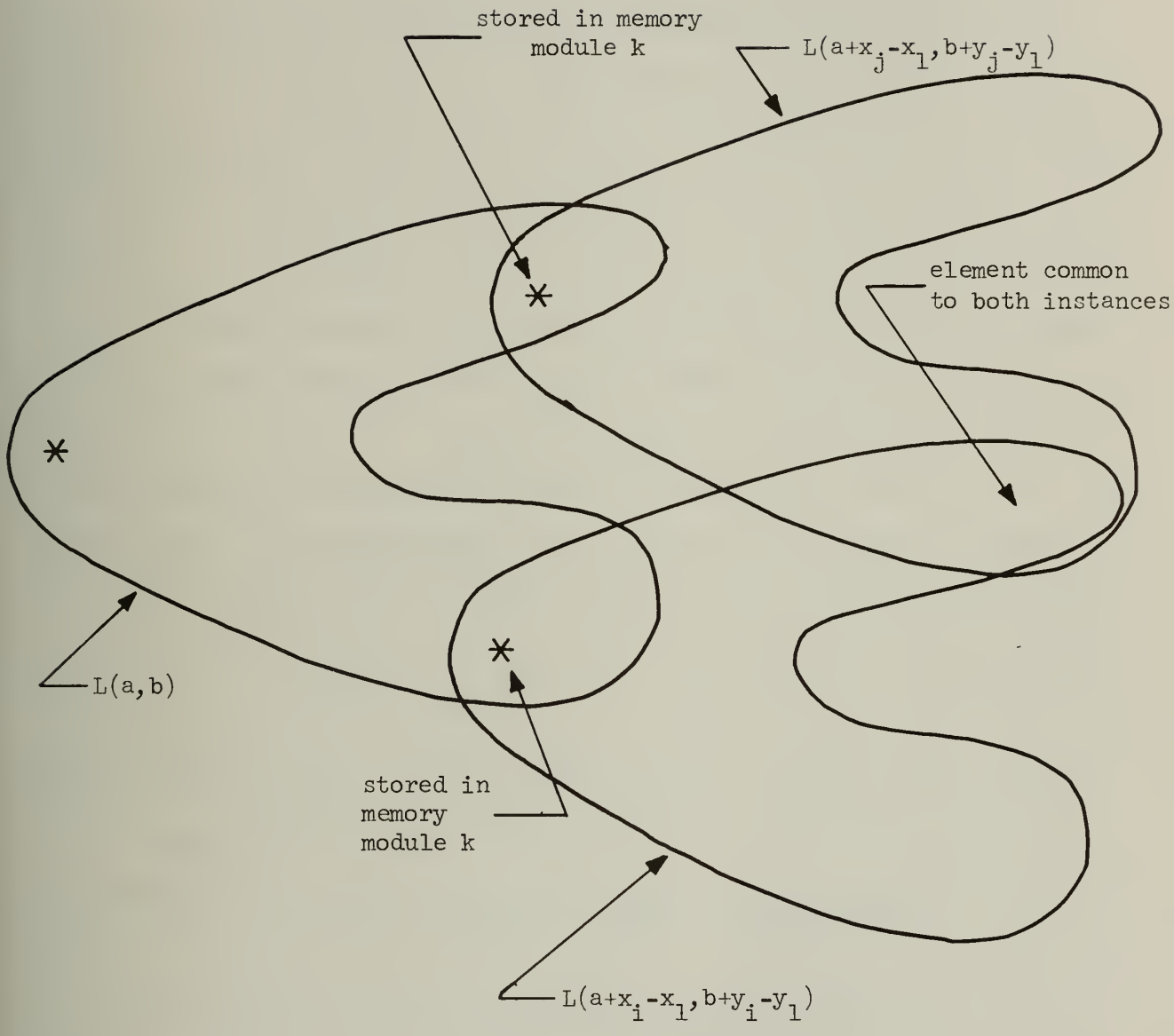
Conversely, suppose $\varphi$ is not a valid skewing scheme for the generalized line, L. Then there is an instance of the generalized line, $L(a,b) = ((a+x_1, b+y_1), (a+x_2, b+y_2), \ldots, (a+x_M, b+y_M))$, the $i^{th}$ and $j^{th}$ components of which, $i \neq j$, are mapped by $\varphi$ into memory k. Then the two instances $L(a+x_j-x_1, b+y_j-y_1)$ and $L(a+x_i-x_1, b+y_i-y_1)$ violate the condition of the theorem. Their designated elements are $(a+x_j-x_1+x_1, b+y_j-y_1+y_1) = (a+x_j, b+y_j)$ and $(a+x_i-x_1+x_1, b+y_i-y_1+y_1) = (a+x_i, b+y_i)$, respectively, which, being the $i^{th}$ and $j^{th}$ components of $L(a,b)$, are mapped into memory k. Furthermore these two instances have an element in common, since the $i^{th}$ component of $L(a+x_j-x_1, b+y_j-y_1)$ and the $j^{th}$ component of $L(a+x_i-x_1, b+y_i-y_1)$ are both $(a+x_j+x_i-x_1, b+y_j+y_i-y_1)$. Figure 10 provides an intuitive picture of the situation. ■

Several remarks are appropriate concerning this theorem. When given a collection of several generalized lines, this theorem can still be used to determine the validity of a skewing scheme by applying it to each generalized line of the collection individually, since the skewing scheme must be valid for each generalized line, independently of its validity for the others in the collection. A second consideration is that this theorem




$L(a_1+x_1-x_j, b_1+y_1-y_j)$ has two components mapped by $\varphi$ to the same memory. Hence $\varphi$ is not valid.

Figure 9: Pictorial Presentation of the Proof of Theorem 4.




$L(a+x_j-x_1, b+y_j-y_1)$ and $L(a+x_i-x_1, b+y_i-y_1)$ have their designated elements stored in memory module k and they have an element in common.

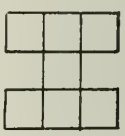
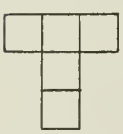
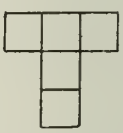
Figure 10: Pictorial Presentation of the Proof of Theorem 4.

is non-constructive. While only providing a method for establishing the validity of a skewing scheme, this theorem will be used in the next section to establish a constructive result.

## 2.3 Existence and Construction of Valid Skewing Schemes when the Number of Memory Modules Equals the Length of the Generalized Line

The results in this section will be developed first for one generalized line, and then extended to collections of generalized lines.

> Definition 9: A generalized line tesselates the plane if and only if there exists a (necessarily infinite) collection of instances of the generalized line, so that every ordered pair in the plane is in one and only one of these instances.

If the realization of a generalized line as a rigid template composed of unit squares is used, then a generalized line tesselates the plane if and only if it can tile the "infinite floor" without gaps or overlapping.† The generalized line, whose realization is  -shaped, tesselates the plane, as can be seen in Figure 11, but the generalized line given by T = ((0,0),(0,1),(0,2),(1,1),(2,1)), whose realization is  -shaped, cannot. Figure 12 illustrates why the  cannot tesselate the plane. Notice that no matter where the first instance is laid down on the plane, there is only one way another instance can be laid down so that the square labeled A is covered and the instances do not overlap. Now the square labeled B cannot be safely covered.

---

†In the literature [5,6] tesselations of the plane normally permit rotations and reflections of the basic shapes. These transformations are excluded in this discussion.

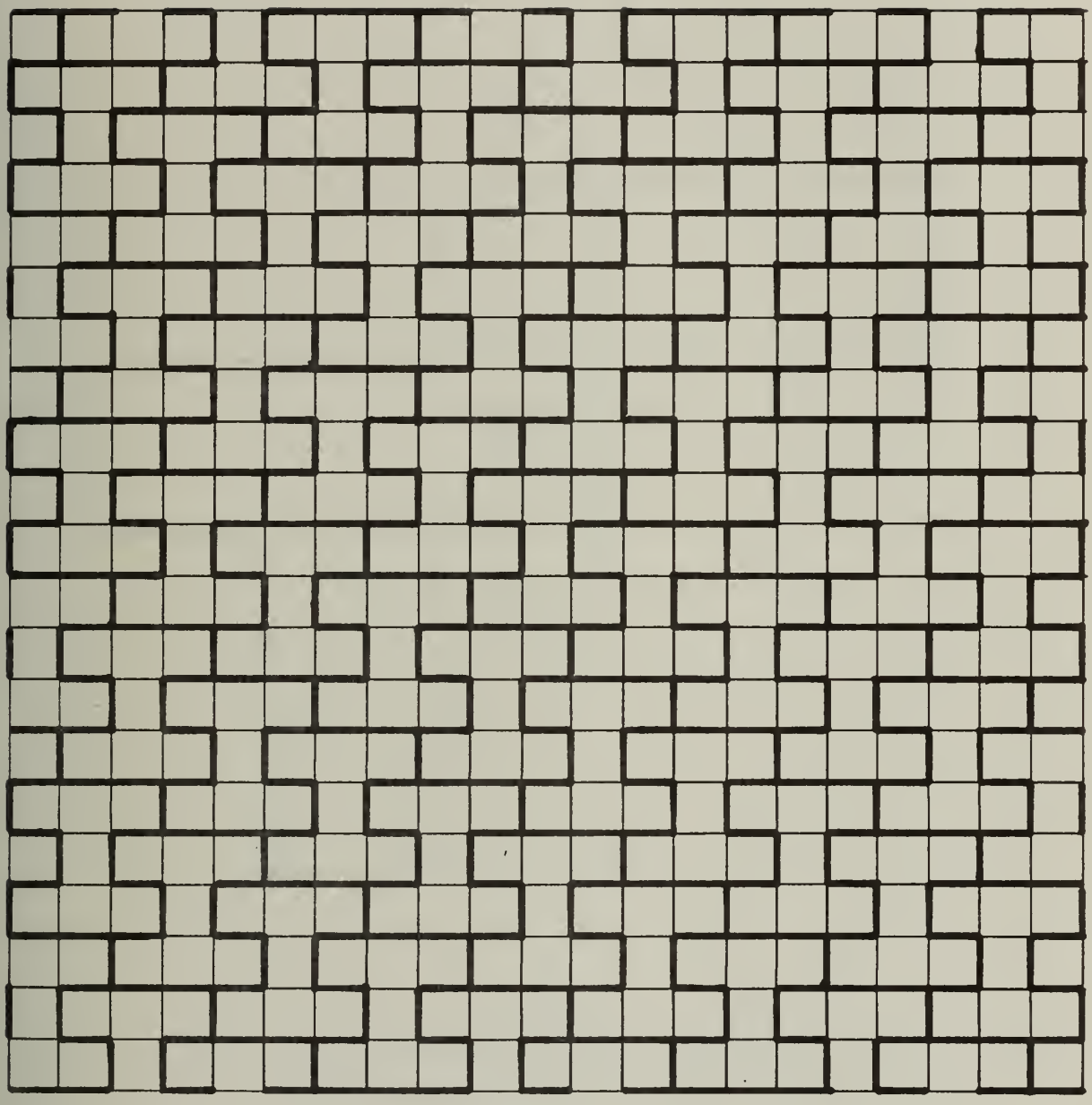


Figure 11: Tesselation of the Plane by a Generalized Line.

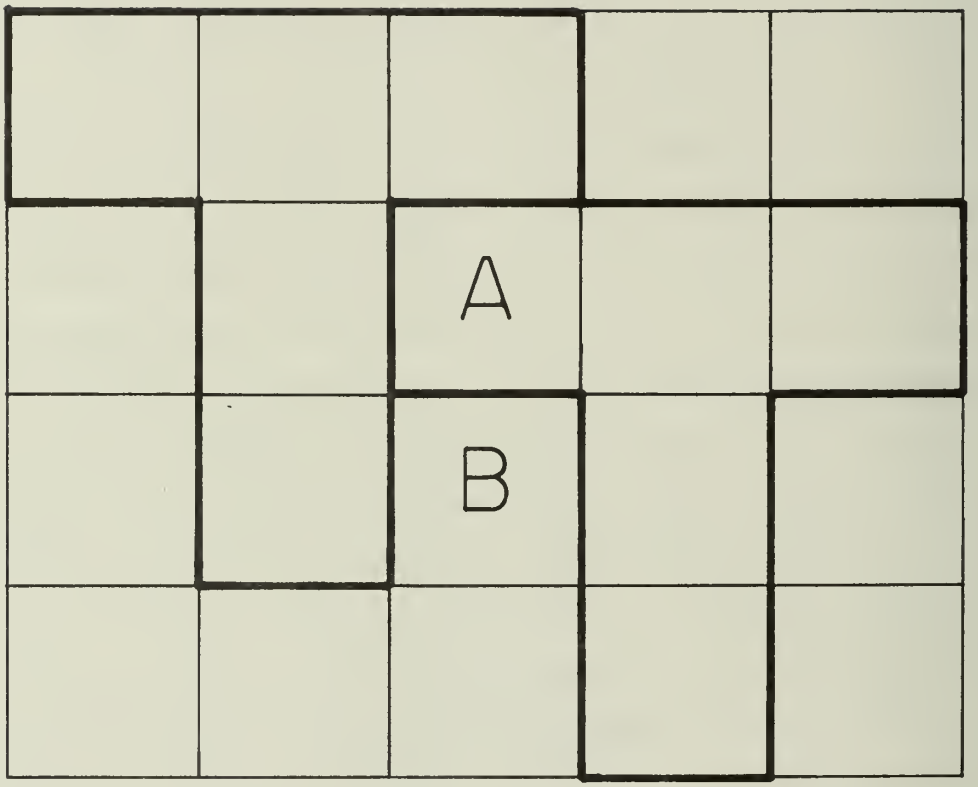



Figure 12: The Generalized Line L = ((0,0),(0,1),(0,2),(1,1),(2,1)) Cannot Tesselate the Plane.

Theorem 5: Given N memory modules and a generalized line of length N, there is a valid skewing scheme for this generalized line if and only if it tesselates the plane.

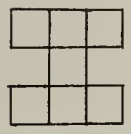
Proof: Suppose that the generalized line tesselates the plane. Define $\varphi$ as follows: $\varphi(i,j) = k$, where $(i,j)$ is the $k+1^{st}$ component of the instance contained in the tesselation which covers the point $(i,j)$. $\varphi$ is well defined since each point in the plane is covered exactly once. The question is: Given any instance, does $\varphi$ map the N components of the instance into distinct memory modules? The reader should note that for $\varphi$ to be valid the answer to the above question must be yes for all instances of the generalized line, including those not contained in the tesselation. Theorem 4 will be applied to show $\varphi$ is valid.

Consider verification of the condition in Theorem 4, when $k = 0$. The set of instances that must be checked for overlap are just those that comprise the given tesselation, by the way $\varphi$ was defined. Since the instances used in a tesselation do not overlap each other, the condition is verified for $k = 0$. Now let k be some other value in $\{0,1,2,\ldots,N-1\}$. Note that every element of the plane stored in memory k is a fixed shift from an element stored in memory zero. To be specific, if the generalized line is $L = ((x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N))$, then every element stored in memory k is at $(x_{k+1}-x_1, y_{k+1}-y_1)$ away from an element stored in memory zero. Conversely, every element in the plane $(x_{k+1}-x_1, y_{k+1}-y_1)$ away from an element stored in memory zero is stored in memory k. Thus when verifying the condition of Theorem 4 for arbitrary k, the set of instances that must be checked for overlap form a tesselation, the tesselation formed by shifting the given tesselation by $(x_{k+1}-x_1, y_{k+1}-y_1)$. Again, since the instances

comprising a tesselation do not overlap, the condition is seen to hold for all k, and, hence, $\varphi$ is valid.

In order to prove the converse suppose $\varphi$ is a valid skewing scheme for the generalized line, L, using N memory modules. Let $T = \{L(i,j) | \varphi(i,j) = 0\}$, i.e. T is the set of instances whose designated elements are mapped by $\varphi$ into memory zero. The claim is that T is a tesselation. The condition in Theorem 4, applied when $k = 0$, guarantees that no two instances of T overlap. Thus, to prove that T is a tesselation it is necessary to show that every point in the plane is a component of some instance contained in T. Suppose not, i.e. there is a point (c,d) such that (c,d) is not a component of any instance in T. Now (c,d) is a component of precisely N instances of L, since there is exactly one instance of the generalized line in which it is the $h^{th}$ component, for $h=1,2,\ldots,N$. Now consider the N distinct designated elements of these N instances. $\varphi$ cannot map any of these N designated elements into memory zero, for it was assumed that (c,d) was not a component of any instance in T. Thus there are two elements in this set, say $(a_1,b_1)$ and $(a_2,b_2)$, so that $\varphi(a_1,b_1) = \varphi(a_2,b_2) = \ell \neq 0$, by the pigeonhole principle. Now the condition of Theorem 4 is violated when $k = \ell$, since $L(a_1,b_1)$ and $L(a_2,b_2)$ will both have their designated elements mapped to memory $\ell$ and they both contain (c,d). This contradicts the validity of $\varphi$. This contradiction arose by assuming T did not cover (c,d). Thus T covers every point in the plane, and T is a tesselation. ■

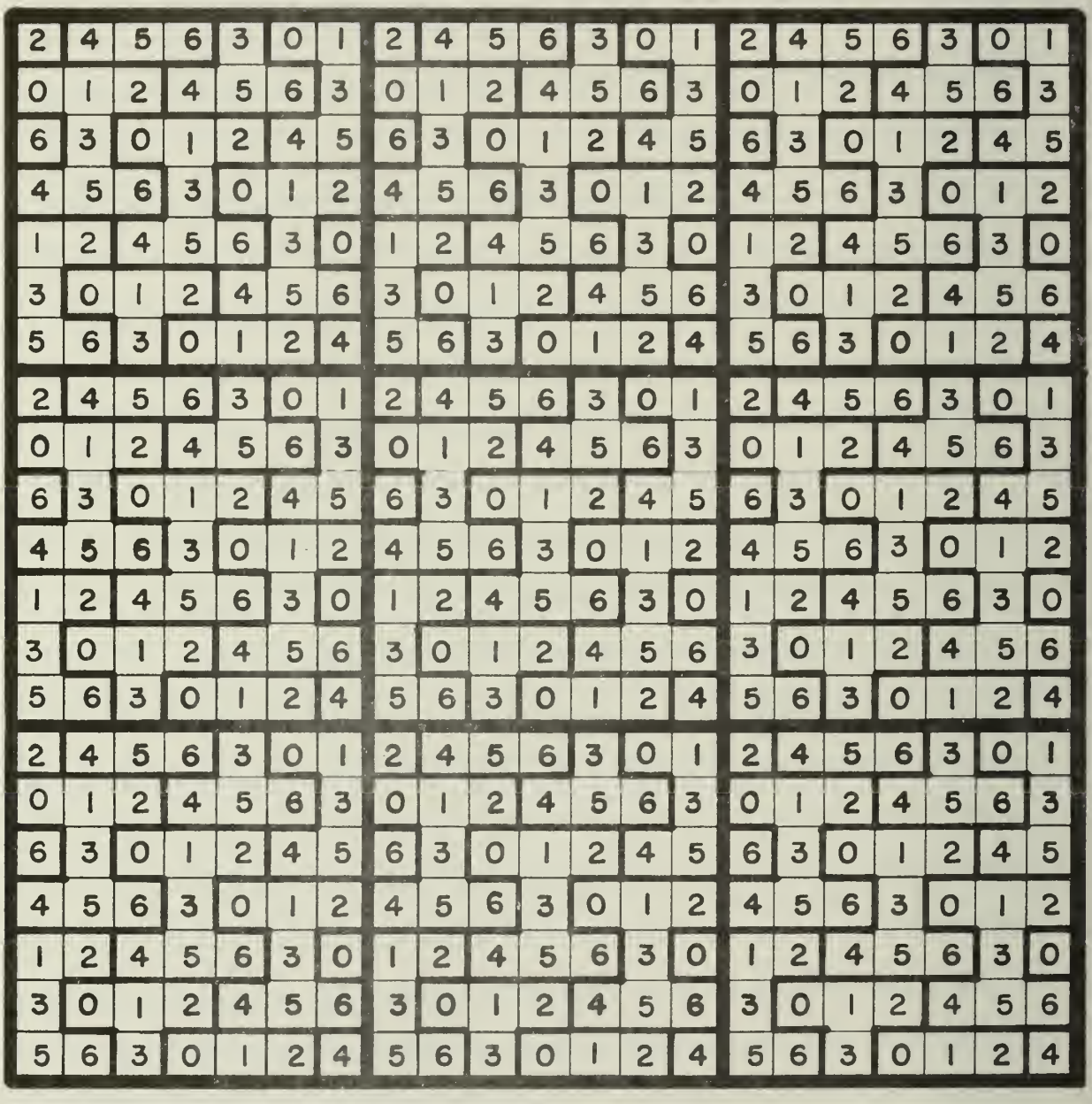
Given a tesselation of the plane by a generalized line, the proof of the theorem shows how a valid skewing scheme can be constructed. This theorem provides a practical means for determining the existence of a valid skewing scheme, since the construction of a tesselation can normally

be done by observation. Figure 13 indicates the skewing scheme resulting from the  -shaped generalized line, and the use of this theorem. The dark heavy lines indicate that in this case the skewing scheme obtained is periodic. More will be said about periodic skewing schemes in other sections.

The result of Theorem 5 can be extended to collections of generalized lines.

> Theorem 6: Given N memory modules and a collection of generalized lines, $\{L_1, L_2, \ldots, L_p\}$, all of length N, then there is a valid skewing scheme for this collection if and only if there exist tesselations of the plane, $T_1, T_2, \ldots, T_p$, such that $T_i$ is a tesselation using $L_i$ and $O_1 = O_2 = \ldots = O_p$, where $O_i$ = {designated elements of the instances of $L_i$ comprising $T_i$}.

Because of Theorem 5, the condition that each generalized line tesselate the plane is clearly needed. An intuitive visualization of the condition on the $O_i$ makes use of Theorem 6 easier, and may aid in understanding the proof. Imagine that each tesselation of the plane, $T_i$, is performed using the rigid template determined by the generalized line, $L_i$, and that each tesselation is done on a separate sheet of clear plastic. In addition, let the designated elements of the instances be marked by asterisks. Then $O_i$ is the set of points on the copy of the plane used for $T_i$ containing an asterisk, and the condition that $O_1 = O_2$ , ... $= O_p$ becomes: When the sheets of clear plastic are overlaid, the asterisks on the sheets of plastic coincide.



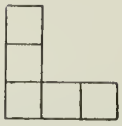

L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2)). The heavy lines indicate that the skewing scheme is periodic.

Figure 13: The Skewing Scheme Resulting From the Use of Theorem 5.

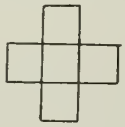
Proof of Theorem 6: Suppose that $\varphi$ is valid for this collection. Then the method of proof used in Theorem 5, applied to each generalized line separately, produces a tesselation $T_\nu$ using $L_\nu$. In addition the tesselation that results by following the construction in that proof yields $O_\nu = \{(i,j) | \varphi(i,j) = 0\}$. Since $O_\nu$ is independent of $\nu$, $O_1 = O_2 = \ldots = O_p$.

To establish the converse suppose tesselations, $T_\nu$, using $L_\nu$, exist, and $O_1 = O_2 = \ldots = O_p$. Then define $\varphi$ by $\varphi(i,j) = k$, where $(i,j)$ is the $k+1^{st}$ component of the instance of $L_1$ contained in $T_1$. This is exactly the same construction employed in the proof of Theorem 5, and so $\varphi$ is well defined and a valid skewing scheme for $L_1$. It must also be shown that $\varphi$ is a valid skewing scheme for $L_2, L_3, \ldots,$ and $L_p$. Pick an arbitrary generalized line, say $L_\nu$. Theorem 4 can be applied to show that $\varphi$ is a valid skewing scheme for $L_\nu$. In verifying that the condition of Theorem 4 holds when $k = 0$, the set of instances that must be examined for common elements are just those instances comprising the tesselation, $T_\nu$. Now consider verifying that the condition of Theorem 4 holds for arbitrary $k$. As in the proof of Theorem 4, the set of instances that must be examined for common elements is obtained by shifting the tesselation. However, the amount of the shift, $(x_{k+1}-x_1, y_{k+1}-y_1)$, is determined by the components of $L_1$, the generalized line used to construct $\varphi$, despite the fact that the condition is being verified for the generalized line, $L_\nu$. Since a rigid translation of a tesselation is a tesselation, and $\nu$ and $k$ were arbitrary, the theorem is proved. ■

A few examples may be helpful in illustrating some of the subtleties that can arise in using this theorem. Consider the generalized lines $L_1 = ((0,0),(1,0),(1,1),(2,0),(2,1))$, whose geometric realization is

, and $L_2 = ((0,0),(1,0),(2,0),(2,1),(2,2))$, whose geometric realization is  . Each of these generalized lines tesselates the plane separately, as Figures 14 and 15 show. Since, these are the only tesselations possible, except for rigid shifting, it is clear that tesselations do not exist for these two generalized lines which can be positioned so their designated elements coincide. Thus there is no valid skewing scheme for the collection $\{L_1, L_2\}$ using only five memories.

A possible question arises from contemplating this example: If another generalized line, $L_\nu'$, which produces the same geometric realization as $L_\nu$, is substituted for $L_\nu$ in the collection $\{L_1, L_2, \cdots, L_p\}$, is it possible a valid skewing scheme will now exist, where before there was no valid skewing scheme? The answer to this question can be important in practice, since it is convenient to think in geometric terms. Fortunately, the substitution described above does not affect the existence of a valid skewing scheme, since tesselations of the plane using $L_\nu'$ appear to the eye as rigid shifts of tesselations of the plane using $L_\nu$. Thus, when actually using this theorem to find valid skewing schemes, it is permissible to just draw pictures, as has been done throughout, and to pick the designated elements arbitrarily.

Figures 14 and 16 illustrate another situation. Both the generalized lines $L_1$, given earlier, and $L_3 = ((0,0),(1,-1),(1,0),(1,1),(2,0))$, whose geometric realization is  , tesselate the plane, and it is clear that when these tesselations are overlaid, their designated elements coincide. Thus Theorem 6 guarantees a skewing scheme using five

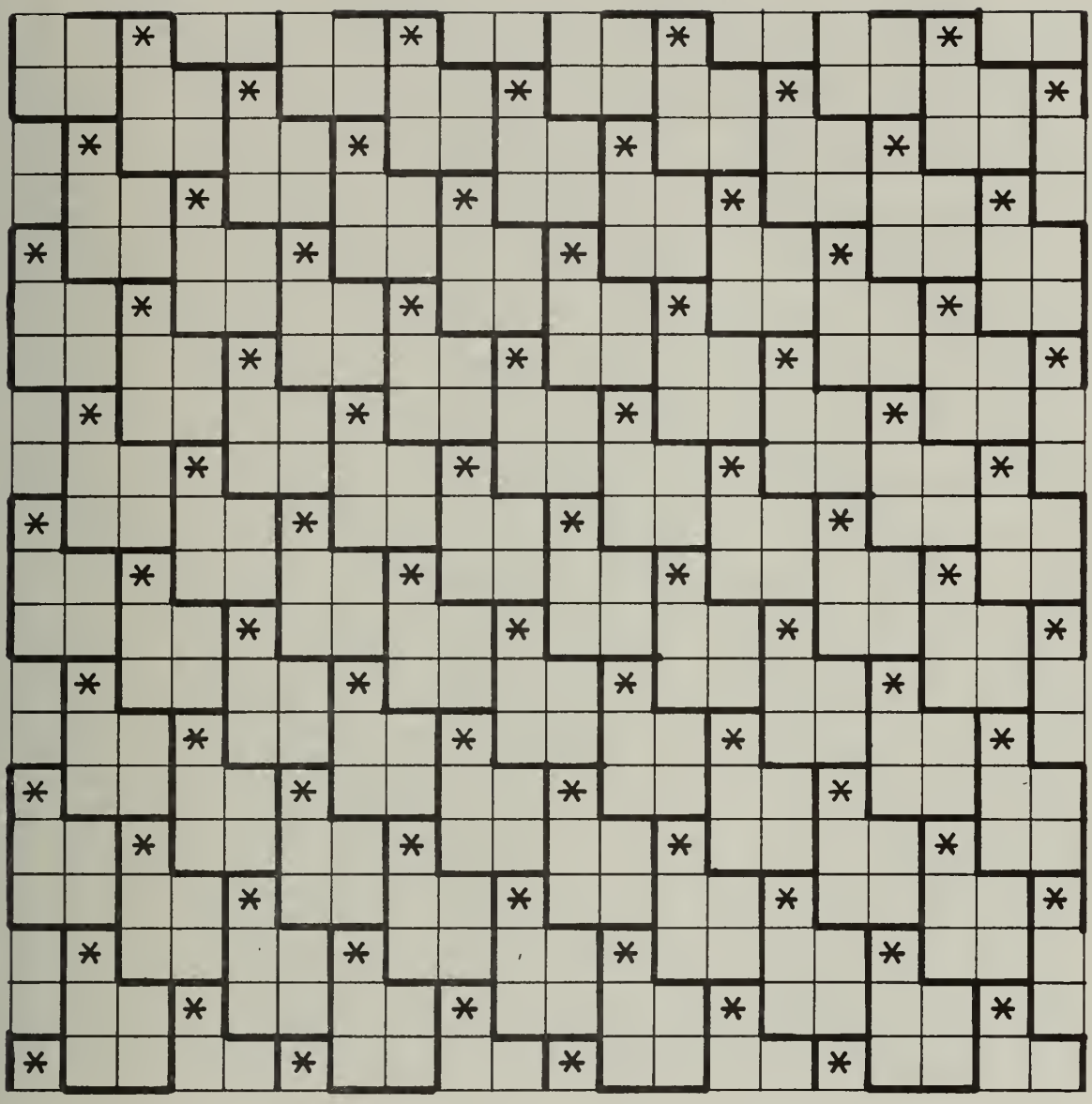


Figure 14: Tesselation of the Plane by the Generalized Line, L = ((0,0),(1,0),(1,1),(2,0),(2,1)).

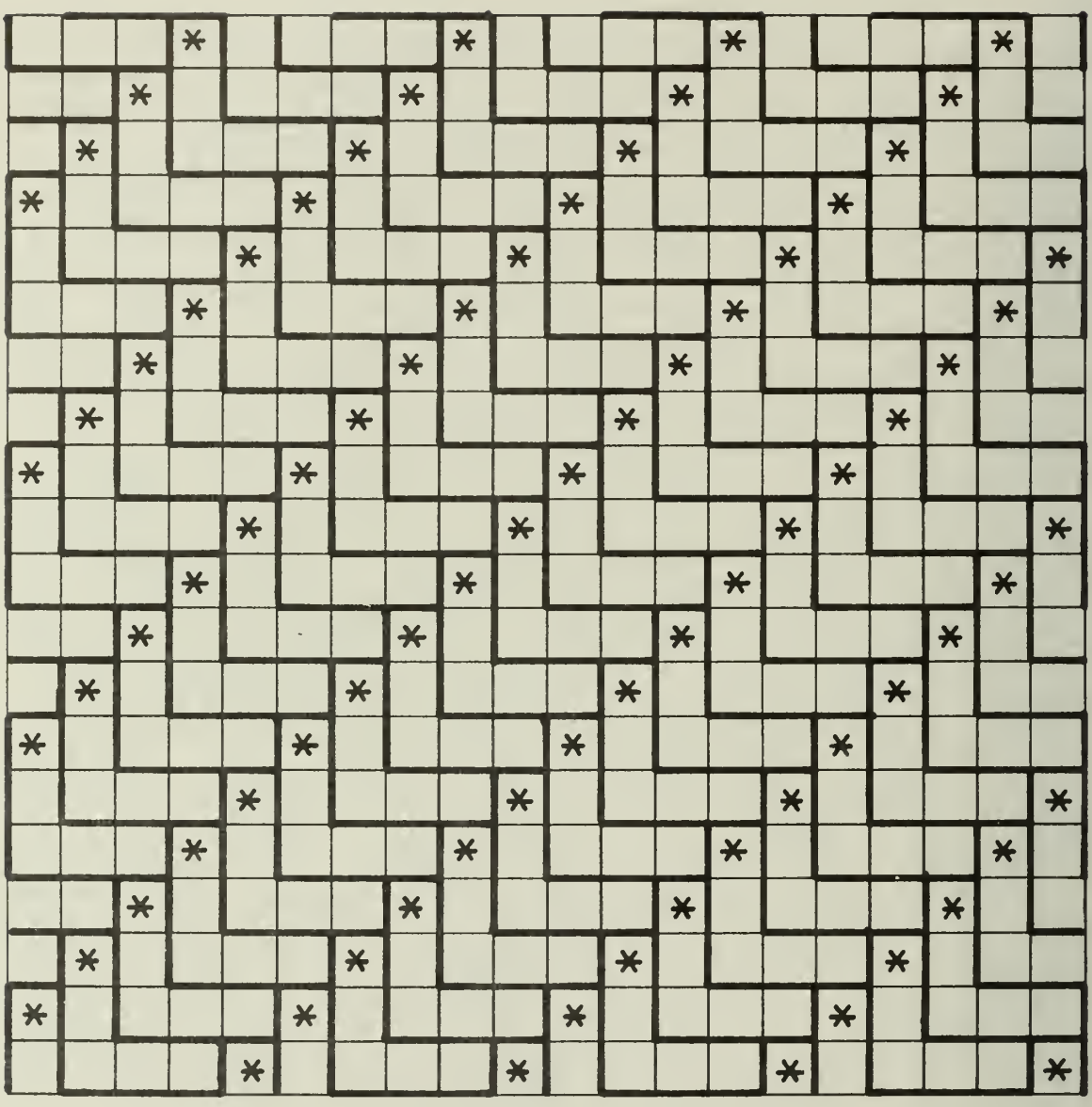


Figure 15: Tesselation of the Plane by the Generalized Line, L = ((0,0),(1,0),(2,0),(2,1),(2,2)).

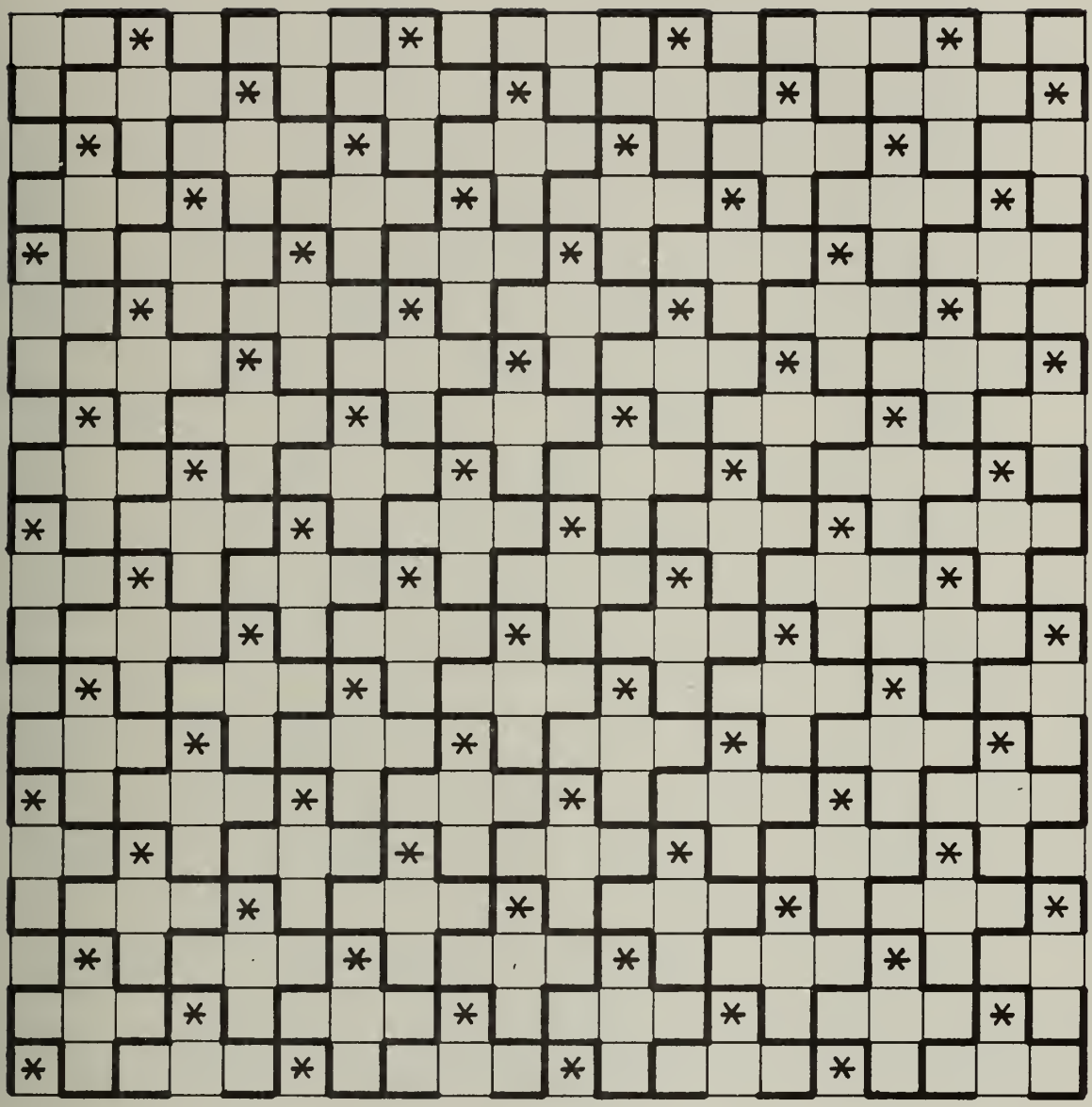


Figure 16: One Possible Tesselation of the Plane by the Generalized Line, L = ((0,0),(1,-1),(1,0),(1,1),(2,0)).

memory modules that allows conflict free access to all instances of either generalized line. However, there is an alternative tesselation for $L_3$, given in Figure 17. If this tesselation had been used, along with the only tesselation for $L_1$, then the incorrect conclusion, that there is no valid skewing scheme for the collection $\{L_1, L_3\}$, might have been drawn. The statement of Theorem 6 only requires the existence of a set of tesselations which also satisfy an additional condition. Thus if more than one tesselation exists for some of the generalized lines in the collection, they must all be tried before concluding that no valid skewing scheme exists.

## 2.4 Existence and Construction of Valid Periodic Skewing Schemes

As was pointed out in Chapter 1, periodic skewing schemes are a valuable subset of all skewing schemes, since, for restricted values of N, a reasonable amount of additional hardware permits address computation by table look-up. For this reason it would be nice if the theorems in the last two sections could be restricted to determine the existence of valid periodic skewing schemes.

The essential idea of the needed alteration is depicted in Figures 13 and 18. Consider the memory storage map, infinite in both directions, defined by a periodic skewing scheme. If the plane is partitioned into $N \times N$ squares, where N is the number of memory modules, then the memory map defined by the skewing scheme is identical within each partition. The bold lines in Figure 13 illustrate this. The main point to be observed is that when an instance of a generalized line extends over one of the partitioning lines, instead of considering the instance to be as in Figure 18(a), it can be considered to be as in Figure 18(b).

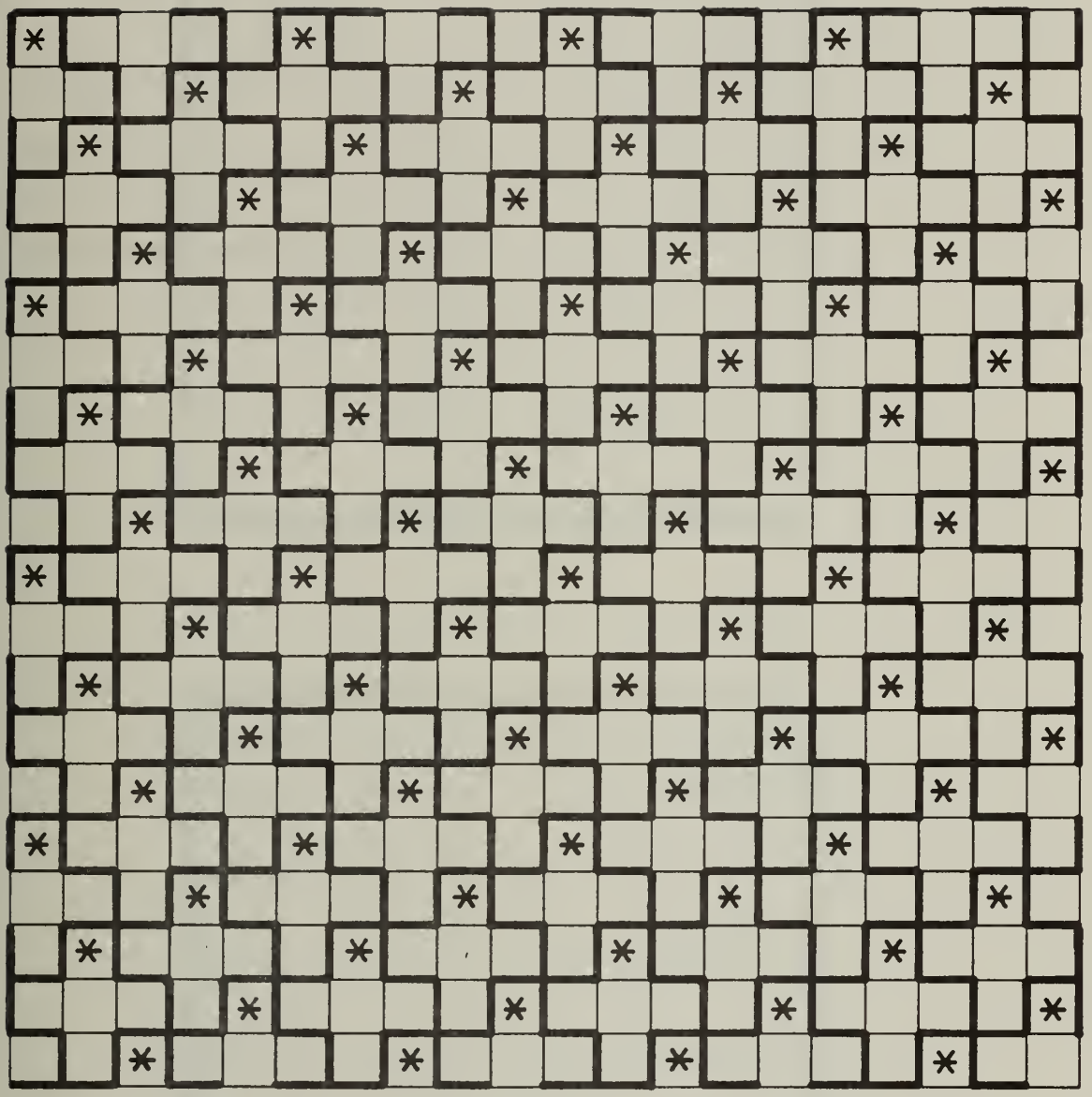
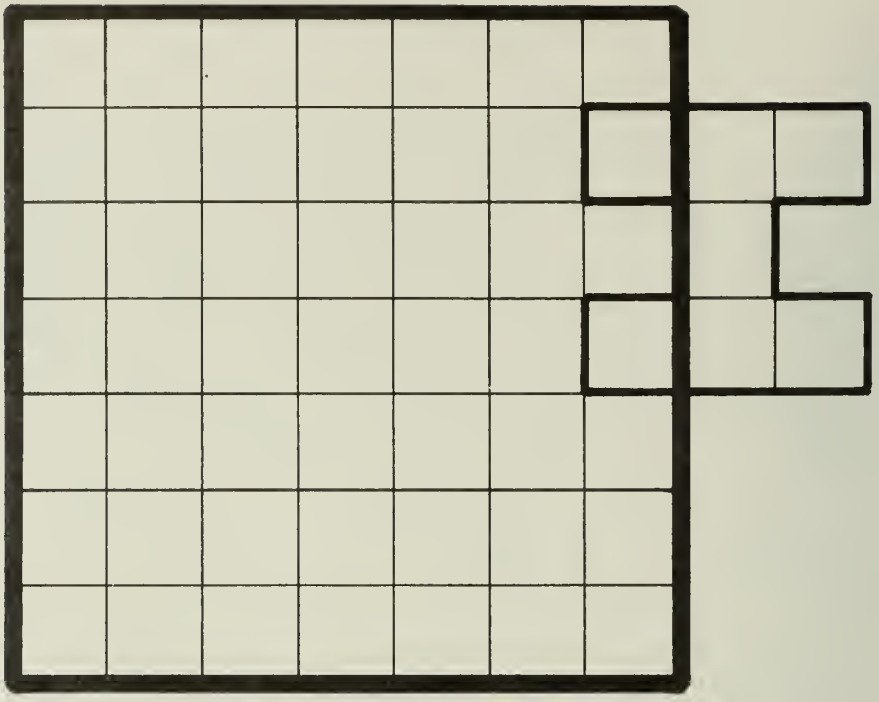
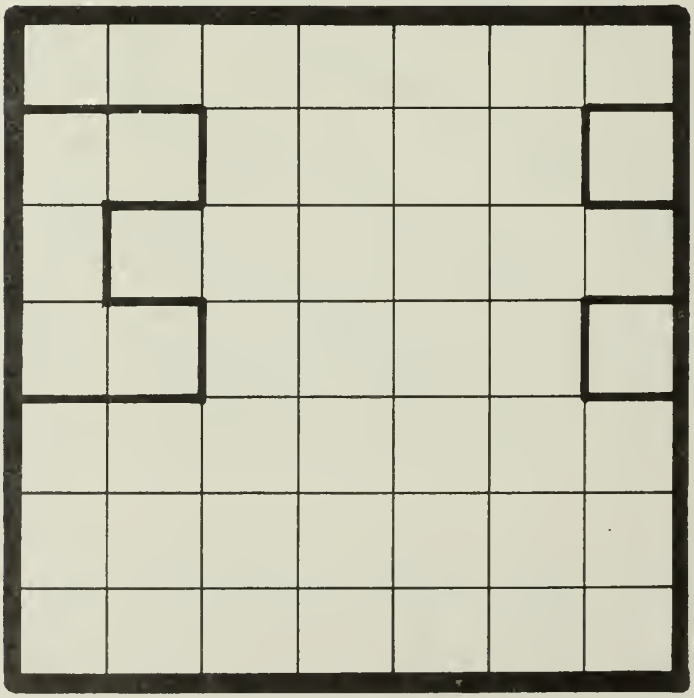


Figure 17: Another Possible Tesselation of the Plane by the Generalized Line, L = ((0,0),(1,-1),(1,0),(1,1),(2,0)).



(a)



(b)

Figure 18: The "Wrap Around" Interpretation of a Generalized Line Used with Periodic Skewing Schemes.

This view is appropriate in determining the validity of a periodic skewing scheme, since in Figure 18(b) the data elements of the instance that are on the left are stored in the same memory modules as the data elements of the instance that extend beyond the partition line in Figure 18(a). This situation has been referred to as "wrapping around." This can occur over horizontal partitioning lines as well. Since there are no special properties used when an instance of a generalized line wraps around, the opposite edges of the $N \times N$ square can be identified, resulting in a torus. The entire problem of finding valid periodic skewing schemes can be recast into the framework of looking for valid skewing schemes on the torus formed by identifying opposite edges of the $N \times N$ square.

Definition 9, and Theorems 4, 5, and 6 carry over to the torus with only minor modification.

> Theorem 7: Given a generalized line of length M, N memory modules, and a periodic skewing scheme, the skewing scheme is valid for this generalized line if and only if the following condition holds for every $k \in \{0,1,2,\ldots,N-1\}$: When all instances of the generalized line on the torus (formed by identifying opposite edges of the $N \times N$ square) are considered, in which the designated element of the instance is mapped into memory k, no two of the instances have an element in common.

> Definition 10: A generalized line <u>tesselates</u> the torus (formed by identifying the opposite edges of an $N \times N$ square) if and only if there exists a collection of

instances of the generalized line, so that every ordered pair on the torus is in one and only one of these instances.

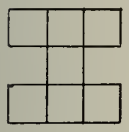
Theorem 8: Given N memory modules and a generalized line of length N, there is a valid periodic skewing scheme for this generalized line if and only if it tesselates the torus (formed by identifying opposite edges of the $N \times N$ square).

Theorem 9: Given N memory modules and a collection of generalized lines, $\{L_1, L_2, \ldots, L_p\}$, all of length N, then there is a valid periodic skewing scheme for this collection if and only if there exists tesselations of the torus (formed by identifying opposite edges of the $N \times N$ square), $T_1, T_2, \ldots, T_p$, such that $T_i$ is a tesselation using $L_i$ and $O_1 = O_2 = \ldots = O_p$, where $O_i$ = {designated elements of the instances of $L_i$ comprising $T_i$}.

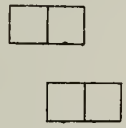
The proofs of these theorems are identical to those of Theorems 4, 5, and 6, only the arithmetic must be done in the residue classes mod N. When drawing pictures to determine the existence of valid periodic skewing schemes only an $N \times N$ square is needed as long as instances extending beyond the bounds of the square are wrapped around.

A few additional comments are in order before closing this section. In working on the torus, as in the plane, the realization of a generalized line, as a rigid template composed of unit squares, need not be connected.

An interesting question which can be asked is: Can the situation be restricted still further to determine the existence of valid linear skewing schemes? The answer to this question is not fully known. Notice that the skewing scheme given in Figure 13 is periodic, but not linear. However, a valid linear skewing scheme does exist for the generalized line whose geometric realization is



-shaped. One such is illustrated in Figure 19. Only the $N \times N$ square is shown, since linear skewing schemes are periodic, and thus only the $N \times N$ square with wrap around need be considered. Other generalized lines, like $L_4 = ((0,0),(0,1),(2,1),(2,2))$, whose geometric realization is the disconnected shape



, provide examples of generalized lines for which valid periodic skewing schemes exist, but for which no valid linear skewing schemes exist. Figure 20 implies the existence of a valid periodic skewing scheme. Trying all possibilities for a and b in $\varphi(i,j) = ai+bj \bmod N$, where, since $N = 4$, a and b need only run over 0, 1, 2, and 3, eliminates the existence of valid linear skewing schemes. The general question of when a valid periodic skewing scheme for a collection of generalized lines implies a valid linear skewing scheme appears quite difficult. Chapter 3 investigates this question for $[x,y]_N$-lines. The general problem is discussed again in Chapter 4.

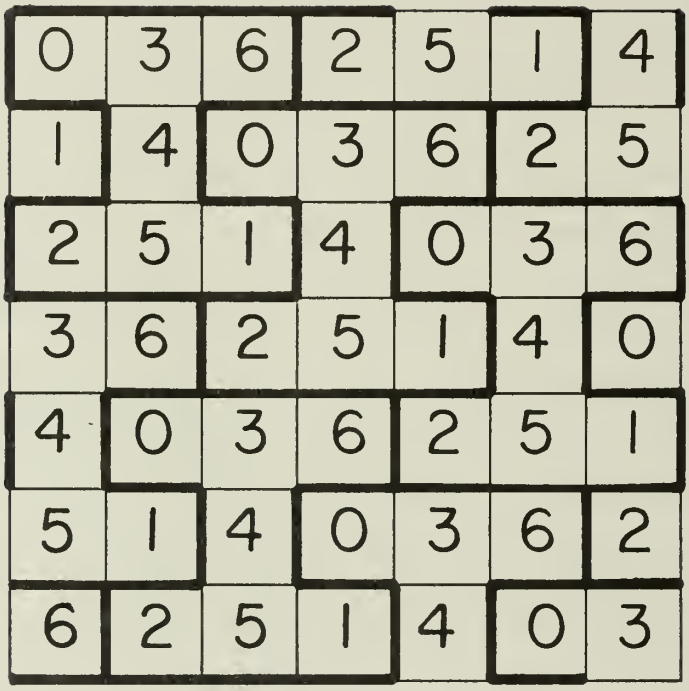



Figure 19: A Valid Linear Skewing Scheme for the Generalized Line, L = ((0,0),(0,1), (0,2),(1,1),(2,0),(2,1),(2,2)).

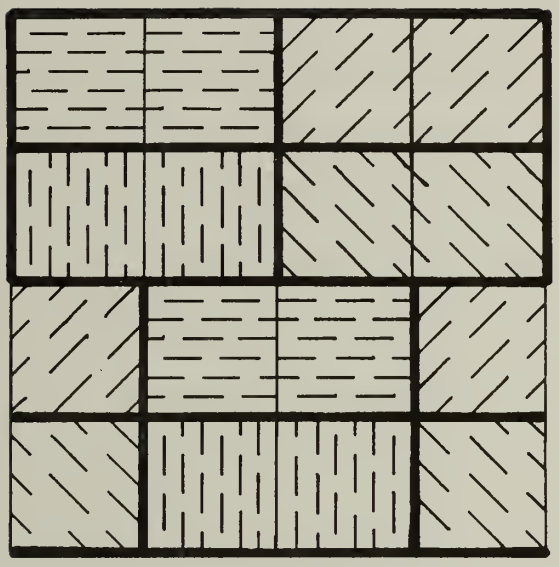


Figure 20: Proof of the Existence of a Periodic Skewing Scheme for L = ((0,0),(0,1),(2,1),(2,2)).

# 3. SPECIAL RESULTS ON $[x,y]_N$-LINES

## 3.1 Introduction

In Chapter 2, geometric conditions were developed which aid in determining if valid skewing schemes exist for collections of generalized lines. In this chapter, only collections of $[x,y]_N$-lines are considered. The highly structured nature of $[x,y]_N$-lines permits additional results to be obtained. The main result, which will be proved over the course of the next several sections, is

> Theorem 10: Given N memory modules and a collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, then there is a valid periodic skewing scheme for the collection if and only if there is a valid linear skewing scheme for the collection.

The if direction is trivial. The only if direction, however, is a rather surprising result, since the number of valid periodic skewing schemes for a collection of generalized lines usually greatly exceeds the number of valid linear skewing schemes. The proof technique is a generalization of an argument used by Pólya for a restricted subcase [13].

## 3.2 Preliminaries

As was discussed in Chapter 2, when dealing with periodic skewing schemes it is convenient to replace the plane with the torus formed by identifying opposite edges of the $N \times N$ square. An instance of an $[x,y]_N$-line can now be viewed as having its first component at $(i,j)$

and with successive components located by going over x and down y on the torus, until a total of N points are generated. Figure 21 illustrates this construction. It might happen that in generating these N points two of them will coincide. If this should occur, then there can be no valid periodic skewing scheme using N memory modules for this $[x,y]_N$-line, since there are two distinct matrix elements, in the same instance of this $[x,y]_N$-line, which must be assigned to the same memory module by any periodic skewing scheme. The condition that must be imposed is simple.

Lemma 1: If $(x_i,y_i,N)^\dagger \neq 1$, then there is no valid periodic skewing scheme using N memory modules for the generalized line, L, the $[x_i,y_i]_N$-line.

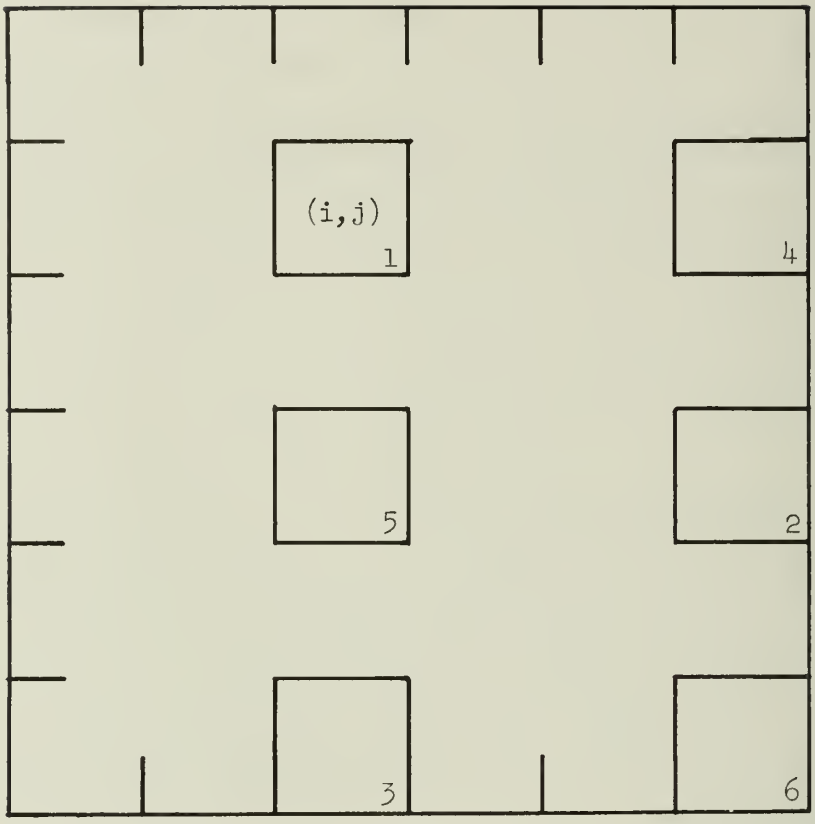
Proof: Suppose $(x_i,y_i,N) = s > 1$. Then $L(c,d) = ((c,d), (c+y_i,d+x_i), \ldots, (c+\frac{N}{s}y_i,d+\frac{N}{s}x_i), \ldots, (c+(N-1)y_i,d+(N-1)x_i))$. Note that $\frac{y_i}{s}$, $\frac{x_i}{s}$, and $\frac{N}{s}$ are integers and that $\frac{N}{s} \leq N-1$. If $\varphi$ is an arbitrary periodic skewing scheme then $\varphi(c,d) = \varphi(c+N\frac{y_i}{s},d+N\frac{x_i}{s}) = \varphi(c+\frac{N}{s}y_i,d+\frac{N}{s}x_i)$. Since (c,d) and $(c+\frac{N}{s}y_i,d+\frac{N}{s}x_i)$ are components of the same instance of L, $\varphi$ is not valid. ■

Thus, given a collection of $[x,y]_N$-lines $\{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, if even one of the $[x_i,y_i]_N$-lines is such that $(x_i,y_i,N) \neq 1$, then there are no valid periodic skewing schemes, using N memory modules, for this collection. $(x_i,y_i,N) = 1$ for $i=1,2,\ldots,I$, is, therefore a necessary condition for the existence of a valid periodic skewing scheme. It is not a sufficient condition, however.

When $(x_i,y_i,N) = 1$ some simplifications are possible. In general, given an arbitrary generalized line, L, of length N, there are $N^2$ distinct instances of L on the torus. However, letting $\hat{L}$ be the

---

† $(x_i,y_i,N)$ is the greatest common divisor of $x_i$, $y_i$ and N.




An example of an instance of an $[x,y]_N$-line on the torus. The order in which the elements were generated is indicated on the figure.

$i = 1, \quad j = 2, \quad x = 3, \quad y = 2, \quad N = 6$

Figure 21: $[x,y]_N$-lines on the Torus.

$[x_i,y_i]_N$-line, on the torus the components of $\hat{L}(c,d)$, $\hat{L}(c+y_i,d+x_i)$, ..., and $\hat{L}(c+(N-1)y_i,d+(N-1)x_i)$ are the same. The ordering of the components within these instances is different, but for purposes of determining the validity of a periodic skewing scheme these N instances can be regarded as one instance. Thus, for $[x,y]_N$-lines, with $(x,y,N) = 1$, the number of instances on the torus is effectively N, instead of $N^2$. Throughout the rest of this chapter an $[x,y]_N$-line, with $(x,y,N) = 1$, will be regarded as having only N instances on the torus. Notice that no two of the N instances have any elements in common and since the torus has $N^2$ elements, every element on the torus is in one and only one instance. It is possible to characterize these N instances.

Lemma 2: Given an $[x,y]_N$-line, with $(x,y,N) = 1$, each of the N instances of the $[x,y]_N$-line can be characterized by an integer in $\{0,1,2,\ldots,N-1\}$, in the following manner: If $(w,z)$ is a component of an instance of the $[x,y]_N$-line, characterize this instance by $xw-yz \bmod N$.

Proof: The proposed characterization is a function from the N instances of the $[x,y]_N$-line to $\{0,1,2,\ldots,N-1\}$. First it must be shown that this is indeed a well-defined function. Let $(w,z)$ and $(w',z')$ be different components of the same instance of the $[x,y]_N$-line. Then $w' \equiv^{\dagger} w+vy$ and $z' \equiv z+vx$, and $xw'-yz' \equiv xw+xvy-yz-yvx \equiv xw-yz$, so, in fact, the mapping defined is a function.

Additionally, the function is a one-to-one correspondence. The pigeonhole principle implies that to see this it suffices to show that for any i there exists w and z such that $xw-yz \equiv i$. It is a well-known

---

†Congruences are mod N, unless otherwise indicated.

result in number theory that given x and y there exists c and d such that $xc-yd = (x,y)$ [7]. Since $((x,y),N) = (x,y,N) = 1$ and the residue classes of numbers relatively prime to N form a group under multiplication, there exists g such that $(x,y) \cdot g \equiv 1$. Hence $xcg-ydg \equiv 1$, and thus $xcgi-ydgi \equiv i$. Letting $w = cgi \bmod N$ and $z = dgi \bmod N$ gives the needed w and z. ■

3.3 The Special Case of a Prime Number of Memory Modules

Theorem 10 is easy to prove if N, the number of memory modules, is a prime.

> Lemma 3: Given N memory modules, N a prime number, and a collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N$-lines $| i=1,2,\ldots,I\}$, then there is a valid periodic skewing scheme for the collection if and only if there is a valid linear skewing scheme for the collection.

Proof: As remarked earlier, the if direction is trivial. If $\varphi$ is a valid periodic skewing scheme then exactly N points on the torus, formed by identifying opposite edges of the $N \times N$ square, will be mapped by $\varphi$ into memory module zero. This is so because exactly one component of each of the N disjoint instances of the $[x_1,y_1]_N$-line must be mapped to zero by $\varphi$. Without loss of generality, the element (0,0) can be assumed to be one of these elements. Consider any other element mapped into memory zero, say the element $(\hat{y},\hat{x})$. Construct the a and b for the linear skewing scheme as follows:

$$\text{if } \hat{y} \equiv 0 \text{ then } a = 1,\ b = 0$$

$$\text{else if } \hat{x} \equiv 0 \text{ then } a = 0,\ b = 1$$

$$\text{else } b = 1,\ a = -\hat{y}^{-1}\hat{x},$$

where $\hat{y}^{-1}$ is the multiplicative inverse of $\hat{y}$ in the field of residue classes mod N. (Note that $\hat{y} \neq 0$ and N is prime.)

The claim is that $\hat{\varphi}(c,d) = ac+bd \bmod N$ is a valid linear skewing scheme. Suppose not, i.e. $\hat{\varphi}(c,d) = \hat{\varphi}(c+\nu y_i, d+\nu x_i)$ for some c,d, $i\epsilon\{1,2,\ldots,I\}$, and $\nu\epsilon\{1,2,\ldots,N-1\}$. This implies $a\nu y_i+b\nu x_i \equiv 0$, but since N is a prime and $\nu \neq 0$, this last equation implies that $ay_i+bx_i \equiv 0$. Three cases will be examined to show that $ay_i+bx_i \equiv 0$ contradicts the validity of $\varphi$.

Case 1: $\hat{y} \equiv 0$. $ay_i+bx_i \equiv 0$ reduces to $y_i \equiv 0$, since a = 1 and b = 0. Since, by Lemma 1, $(x_i,y_i,N) = 1$, $x_i \neq 0$. It follows that $x_i^{-1}$ exists in the field of residue classes mod N, since N is prime and $x_i$ is non-zero. Thus $(\hat{y},\hat{x}) \equiv (0,\hat{x}) \equiv (0,\hat{x}\hat{x}_i^{-1}x_i) \equiv (0+\hat{x}\hat{x}_i^{-1}y_i, 0+\hat{x}\hat{x}_i^{-1}x_i)$. Now (0,0) and $(0+\hat{x}\hat{x}_i^{-1}y_i, 0+\hat{x}\hat{x}_i^{-1}x_i)$ are distinct components of the same instance of the $[x_i,y_i]_N$-line, because $\hat{x}\hat{x}_i^{-1}$ is non-zero. They are both, however, mapped by $\varphi$ to memory zero, contradicting the validity of $\varphi$.

Case 2: $\hat{x} \equiv 0$. $ay_i+bx_i \equiv 0$ reduces to $x_i \equiv 0$. Here $y_i \neq 0$, and, in a manner similar to case 1, $(\hat{y},\hat{x}) \equiv (\hat{y},0) \equiv (\hat{y}\hat{y}_i^{-1}y_i,0) \equiv (0+\hat{y}\hat{y}_i^{-1}y_i, 0+\hat{y}\hat{y}_i^{-1}x_i)$. This is a similar situation to that encountered in case 1.

Case 3: $\hat{x} \neq 0$, $\hat{y} \neq 0$. $ay_i+bx_i \equiv 0$ becomes $-\hat{y}^{-1}\hat{x}y_i+x_i \equiv 0$. Now $y_i \neq 0$, for $y_i \equiv 0$ implies $x_i \equiv 0$ and then $(x_i,y_i,N) \neq 1$, contrary to the requirement established by Lemma 1. But $y_i \neq 0$ implies $y_i^{-1}$ exists in the field of residue classes mod N. Thus $\hat{x} \equiv \hat{y}\hat{y}_i^{-1}x_i$ and $(\hat{y},\hat{x}) \equiv (\hat{y}\hat{y}_i^{-1}y_i, \hat{y}\hat{y}_i^{-1}x_i) \equiv (0+\hat{y}\hat{y}_i^{-1}y_i, 0+\hat{y}\hat{y}_i^{-1}x_i)$, giving rise to the same contradiction as in case 1.

Thus, the assumption that $\hat{\varphi}(c,d) = ac+bd \bmod N$ is not a valid linear skewing scheme led to the conclusion that $\varphi$ is not a valid periodic skewing scheme, contrary to the hypothesis of the lemma. ■

It is now possible to precisely characterize when valid periodic skewing schemes exist for a collection of $[x,y]_N$-lines, if the number of memory modules is a prime.

Lemma 4: Given N memory modules, N a prime number, and a collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N$-lines$|$ $i=1,2,\ldots,I\}$, then there exist a valid periodic skewing scheme for this collection if and only if $(x_i,y_i,N) = 1$, for $i=1,2,\ldots,I$, and either $I \leq N$ or for all non-zero choices for $g_i$, $i=1,2,\ldots,N+1$ and any permutation, $\sigma$, of $(1,2,\ldots,I)$, it is not the case that $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, $(x_{\sigma(2)},y_{\sigma(2)}) \equiv (g_2,g_2)$, $\ldots$, $(x_{\sigma(i)},y_{\sigma(i)}) \equiv (g_i(i-1),g_i)$, $\ldots$, $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$.

Proof: Lemma 3 shows that it is sufficient to prove that the conditions stated above are precisely those needed to guarantee the existence of valid linear skewing schemes for the collection of $[x,y]_N$-lines. The need for the condition $(x_i,y_i,N) = 1$ for $i=1,2,\ldots,I$ has already been discussed.

The only if direction: Suppose $I \geq N+1$ and there exists non-zero $g_i$ and a permutation, $\sigma$, so that $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, $\ldots$, $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$. It will be shown that for any a and b, $\varphi(c,d) = ac+bd \bmod N$ is not a valid skewing scheme.

Case 1: $b \equiv 0$. $ay_{\sigma(N+1)}+bx_{\sigma(N+1)} \equiv a \cdot 0 + 0 \cdot g_{N+1} \equiv 0$. Thus $(ay_{\sigma(N+1)}+bx_{\sigma(N+1)},N) \neq 1$, and, by Theorem 3, $\varphi$ is not a valid skewing scheme.

Case 2: $b \not\equiv 0$. Because N is a prime number, $(ay_i+bx_i, N) = 1$ for all i if and only if $ay_i+bx_i \not\equiv 0$ for all i. Also $ay_i+bx_i \not\equiv 0$ for all i if and only if $ab^{-1}y_i+x_i \not\equiv 0$ for all i. Now $ab^{-1} \equiv j$ for some $j\epsilon\{0,1,2,\ldots,N-1\}$. Let $k \equiv -j$. Then $ab^{-1}y_{\sigma(k+1)}+x_{\sigma(k+1)} \equiv ab^{-1}g_{k+1}+g_{k+1}k \equiv jg_{k+1}-jg_{k+1} \equiv 0$, where $(x_{\sigma(k+1)},y_{\sigma(k+1)}) \equiv (g_{k+1}k,g_{k+1})$ was used. Thus $ay_i+bx_i \equiv 0$ for some i and Theorem 3 implies $\varphi$ is not valid.

The if direction: $(x_i,y_i,N) = 1$ for i=1,2,...,I and suppose, for the moment, $I \geq N+1$, but for all non-zero choices for $g_i$ and any permutation, $\sigma$, of (1,2,...,I), it is not the case that $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, ..., $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$. A valid linear skewing scheme will be defined for the collection of $[x,y]_N$-lines.

Case 1: Suppose that there exists $j\epsilon\{0,1,2,\ldots,N-1\}$, such that $(x_i,y_i) \not\equiv (g_{j+1}j,g_{j+1})$ for all i and all non-zero choices of $g_{j+1}$, i.e. the reason a permutation cannot be constructed with $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, ..., $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$ is that there is no possible choice for $\sigma(j+1)$. (Note that when attempting to construct permutations for which $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, ..., $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$, $(x_i,y_i)$ can be congruent to only one of $(0,g_1)$, $(g_2,g_2)$, ..., $(g_N(N-1),g_N)$, and $(g_{N+1},0)$.) Take a = -j and b = 1. Now if $(ay_k+bx_k,N) \not\equiv 1$ for some k, then $ay_k+bx_k \equiv 0$, since N is prime, and hence $-jy_k+x_k \equiv 0$. Then $(x_k,y_k) \equiv (jy_k,y_k)$ contrary to the assumption that $(x_i,y_i) \not\equiv (g_{j+1}j,g_{j+1})$ for any i and any non-zero $g_{j+1}$, since $g_{j+1}$ can be taken to be $y_k$. Hence, $(ay_i+bx_i,N) = 1$ for all i and $\varphi(c,d) = ac+bd \bmod N$ is valid.

Case 2: For any i and any non-zero $g_{N+1}$, $(x_i,y_i) \not\equiv (g_{N+1},0)$. This is similar to case 1, and occurs when there is no possible choice for $\sigma(N+1)$ for which $(x_{\sigma(1)},y_{\sigma(1)}) = (0,g_1)$, ..., $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$. Take a = 1 and b = 0. Again, suppose $(ay_k+bx_k,N) \neq 1$ for some k, then $ay_k+bx_k \equiv 0$ reduces to $y_k \equiv 0$, and $(x_k,y_k) \equiv (x_k,0) \equiv (g_{N+1},0)$ where $g_{N+1}$ is taken as $x_k$. This contradicts the assumption that $(x_i,y_i) \not\equiv (g_{N+1},0)$ for all i and all non-zero choices of $g_{N+1}$. Thus $\varphi(c,d) = ac+bd \bmod N$ is valid.

Since these two cases exhaust all the reasons why a permutation, $\sigma$, cannot be constructed so that for some non-zero choices of $g_i$ $(x_{\sigma(1)},y_{\sigma(1)}) \equiv (0,g_1)$, ..., $(x_{\sigma(N)},y_{\sigma(N)}) \equiv (g_N(N-1),g_N)$, and $(x_{\sigma(N+1)},y_{\sigma(N+1)}) \equiv (g_{N+1},0)$, there is a valid linear skewing scheme for the collection of $[x,y]_N$-lines.

If $I \leq N$ the pigeonhole principle gives the same two cases, since there will either exist a $j\in\{0,1,\ldots,N-1\}$ such that for any $i\in\{1,2,\ldots,I\}$ and any non-zero choice of $g_{j+1}$, $(x_i,y_i) \not\equiv (g_{j+1}j,g_{j+1})$ or for any $i\in\{1,2,\ldots,I\}$ and any non-zero choice of $g_{N+1}$, $(x_i,y_i) \not\equiv (g_{N+1},0)$. ■

## 3.4 Generalization to Composite N

To complete the proof of Theorem 10 it suffices to prove two statements:

1. Given N, the number of memory modules, N a composite number, a collection of $[x,y]_N$-lines, $C = \{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, and a valid periodic skewing scheme, $\varphi$, for this collection, using N memory modules, then if p is a prime factor of N, there is a valid periodic skewing

scheme, $\varphi'$, using only p memory modules, for the collection of $[x,y]_p$-lines, $C' = \{[x_i,y_i]_p\text{-lines} \mid i=1,2,\ldots,I\}$.†

2. Given a collection of $[x,y]_M$-lines, $S = \{[x_i,y_i]_M\text{-lines} \mid i=1,2,\ldots,I\}$, and a valid linear skewing scheme, $\varphi$, for S, using M memory modules, and given a valid linear skewing scheme, $\varphi'$, for $S' = \{[x_i,y_i]_{M'}\text{-lines} \mid i=1,2,\ldots,I\}$, using M' memory modules, then there is a valid linear skewing scheme, $\varphi''$, using MM' memory modules, for the collection of $[x,y]_{MM'}$-lines, $S'' = \{[x_i,y_i]_{MM'}\text{-lines} \mid i=1,2,\ldots,I\}$.

To see that this is sufficient note that given a valid periodic skewing scheme, using N memory modules, for the collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, statement 1 above, implies valid periodic skewing schemes for each prime factor of N. Lemma 3 then implies the existence of valid linear skewing schemes for this collection for each prime factor of N, and, finally, statement 2 implies existence of a valid linear skewing scheme for this collection using N memories. Because of Theorem 3, statement 2 is equivalent to

Lemma 5: Given a collection of ordered pairs, $\{(x_i,y_i) \mid i=1,2,\ldots,I\}$, and given M, M', a, b, a', and b', such that $(ay_i+bx_i,M) = (a'y_i+b'x_i,M') = 1$, for $i=1,2,\ldots,I$, then there exists a" and b" such that $(a''y_i+b''x_i,MM') = 1$, for $i=1,2,\ldots,I$.

---

† Note that the $x_i$ and $y_i$ are the same in both C and C', but the generalized lines are different, since they have different length.

Proof: The proof is constructive. Let $\pi$ be the product of those prime numbers which divide M, but do not divide M'. Let $\pi'$ be the product of those prime numbers which divide M', but do not divide M. Finally, let $\rho$ be the product of those primes which divide both M and M'. For the sake of definiteness, in calculating $\pi$, $\pi'$ and $\rho$ include a prime factor in the product only once, even if it appears in the prime factorization of M and/or M' several times. Also if there are no prime factors from which to calculate $\pi$, $\pi'$ or $\rho$ set $\pi$, $\pi'$ or $\rho$, as the case dictates, to one.

Define $a'' = a\pi'\rho+a'\pi$ and $b'' = b\pi'\rho+b'\pi$. The claim is that $(a''y_i+b''x_i, MM') = 1$ for $i=1,2,\ldots,I$. Suppose not, i.e. for some i, there is a prime, p, for which $p|a''y_i+b''x_i$ and $p|MM'$. If $p|a''y_i+b''x_i$, then $p|a\pi'\rho y_i+a'\pi y_i+b\pi'\rho x_i+b'\pi x_i$. By the definitions of $\pi$, $\pi'$ and $\rho$, p divides exactly one of them. Assume $p|\pi$. Since, $p|\pi$ implies $p|a'\pi y_i+b'\pi x_i$, it can be concluded that $p|a\pi'\rho y_i+b\pi'\rho x_i$. Since p is a prime and $p \nmid \pi'$ and $p \nmid \rho$, it must happen that $p|ay_i+bx_i$. However, $p|\pi$ implies $p|M$, so $(ay_i+bx_i, M) \neq 1$, contrary to the hypothesis of the theorem. Thus if $p|a''y_i+b''x_i$ and $p|MM'$, then $p \nmid \pi$. By similar arguments $p \nmid \pi'$ and $p \nmid \rho$. But this is impossible. Hence, $a''y_i+b''x_i$ and MM' have no prime factors in common, i.e. $(a''y_i+b''x_i, MM') = 1$ for $i=1,2,\ldots,I$. ■

To complete the proof of Theorem 10, it is, therefore, sufficient to prove statement 1, above. Statement 1 will be proved by contradiction. To this end, suppose a collection of $[x,y]_p$-lines, $C = \{[x_i,y_i]_p\text{-lines} \mid i=1,2,\ldots,I\}$, is given, p is a prime number, there is no valid periodic skewing scheme for C, using p memory modules, and there is a valid periodic

skewing scheme, using N memory modules for the collection of $[x,y]_N$-lines, $C' = \{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, where $p|N$.

Some additional technical lemmas are useful.

Lemma 6:

$$\begin{bmatrix} p-1 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 1^{-1} & 1^{-2} & \cdots & 1^{-(p-2)} & 1^{-(p-1)} \\ 0 & 2^{-1} & 2^{-2} & \cdots & 2^{-(p-2)} & 2^{-(p-1)} \\ & & & \vdots & & \\ 0 & (p-1)^{-1} & (p-1)^{-2} & \cdots & (p-1)^{-(p-2)} & (p-1)^{-(p-1)} \end{bmatrix} \times$$

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 2 & \cdots & (p-1) \\ & & & \vdots & \\ 0 & 1^{p-2} & 2^{p-2} & \cdots & (p-1)^{p-2} \\ 0 & 1^{p-1} & 2^{p-1} & \cdots & (p-1)^{p-1} \end{bmatrix} = (p-1)I\,,$$

where all calculations are done in the field of residue classes mod p, p a prime number.

Proof: For convenience the second matrix will be denoted by $M_p$ and the first by $M_p'$. Consider the first row of the product matrix. Clearly, the first row of $M_p'$ times the first column of $M_p$ is p-1. The first row of $M_p'$ times any other column of $M_p$ is zero, since it is $(p-1)\cdot 1+0\cdot i+\ldots+0\cdot i^{p-2}+1\cdot i^{p-1} = (p-1)+i^{p-1}$. Fermat's theorem [7] states that, if $i \neq 0$,† $i^{p-1} = 1$. Thus $p-1+i^{p-1} = p-1+1 = p = 0$. Thus the first

†Note that all equations are in the field of residue classes mod p, in particular, = mean $\equiv_p$.

row of the product matrix is [p-1 0 0 ... 0]. Now consider any other row of $M'_p$, $[0\ i^{-1}\ i^{-2}\ \ldots\ i^{-(p-1)}]$, times any column of $M_p$, $[1\ j\ j^2\ \ldots\ j^{p-1}]^T$. The product is $0+i^{-1}j+i^{-2}j^{-2}+\ldots+i^{-(p-1)}j^{p-1} = (i^{-1}j)+(i^{-1}j)^2+\ldots+(i^{-1}j)^{p-1}$.

In general $(x+x^2+\ldots+x^{p-1})(x-1) = x^p-x$, but since p is prime, Fermat's theorem implies $x^p = x$ for any x. Thus $(x+x^2+\ldots+x^{p-1})(x-1) = 0$. But in a field, the product of two numbers is zero if and only if at least one of them is zero. Thus if $x \neq 1$, $x+x^2+\ldots+x^{p-1} = 0$, and, clearly, if $x = 1$, $x+x^2+\ldots+x^{p-1} = p-1$. Replacing x by $i^{-1}j$, and noting $i^{-1}j = 1$ if and only if $i = j$, gives that the off-diagonal elements of the product matrix are zero and the diagonal elements are p-1. Thus $M'_p \times M_p = (p-1)I$. ■

Corollary 1: $\det(M_p) \neq 0$, where the calculations are performed in the field of residue classes mod p, p a prime number.

Proof: $(p-1)M'_p$ is a left inverse for $M_p$, since $(p-1)^2 = p^2-2p+1 = 1$. But if a matrix has a left inverse, the left inverse is a two-sided inverse, and the matrix has a non-zero determinant [8]. ■

Corollary 2: $$\sum_{i=0}^{p-1} i^r \equiv_p \begin{cases} p-1 & \text{if } r = p-1 \\ 0 & \text{if } r = 0,1,2,\ldots,p-2 \end{cases},$$

where $0^0 = 1$, by convention.

Proof: While, in general, matrix multiplication does not commute, $M'_p \times M_p = M_p \times M'_p$, since $M'_p$ is (almost) the inverse of $M_p$. Note that the last column of $M'_p$ is all ones, since $i^{-(p-1)} = (i^{-1})^{p-1} = 1$, by Fermat's theorem. Thus $(\sum_{i=0}^{p-1} i^r) \bmod p$ is just the $r+1^{st}$ row of $M_p$ times the last column of $M'_p$. Since $M_p \times M'_p = (p-1)I$, the last column of the product matrix is $[0\ 0\ \ldots\ 0\ p-1]^T$, and the corollary is proved. ■

Lemma 7: $$\sum_{i=0}^{p^e-1} i^r \equiv_{p^e} \begin{cases} -p^{e-1} & \text{if } r = p-1 \\ 0 & \text{if } r = 0,1,\ldots,p-2 \end{cases}, \text{ for } e \geq 1. \quad (1)$$

Proof: The proof is by induction on e. For the basis case, e = 1, formula (1) reduces to $$\sum_{i=0}^{p-1} i^r \equiv_p \begin{cases} -p^0 = -1 = p-1 & \text{if } r = p-1 \\ 0 & \text{if } r = 0,1,2,\ldots,p-2 \end{cases}.$$ This is just Corollary 2.

Therefore, assume that the result is true for e' = e-1. Since formula (1) is clearly true for r = 0, r will be assumed greater than zero throughout the remainder of the proof. Now,

$$\sum_{\ell=0}^{p^e-1} \ell^r = \sum_{j=0}^{p^{e-1}-1} \sum_{i=0}^{p-1} (jp+i)^r = \sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} (jp+i)^r . \quad (2)$$

Expanding $(jp+i)^r$ by the binomial expansion and rearranging the terms in the sum gives

$$\sum_{\ell=0}^{p^e-1} \ell^r = \sum_{k=0}^{r} \sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k . \quad (3)$$

Isolate an inner sum,

$$\sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k = \binom{r}{k} p^{r-k} i^k \sum_{j=0}^{p^{e-1}-1} j^{r-k} . \quad (4)$$

By the induction assumption,

$$\sum_{j=0}^{p^{e-1}-1} j^{r-k} \equiv_{p^{e-1}} \begin{cases} -p^{e-2} & \text{if } r-k = p-1 \\ 0 & \text{if } r-k = 0,1,\ldots,p-2 \end{cases}.$$

Thus,

$$\sum_{j=0}^{p^{e-1}-1} j^{r-k} = \begin{cases} -p^{e-2}+\alpha_{r,k}p^{e-1} & \text{if } r-k = p-1 \\ \alpha_{r,k}p^{e-1} & \text{if } r-k = 0,1,\ldots,p-2 \end{cases} \tag{5}$$

for some $\alpha_{r,k}$, integer. By substituting (5) into (4) it is possible to determine the value of the sums in (3).

Case 1: Middle terms of the binomial expansion, $r > k > 0$. In this case $1 \leq r-k < p-1$. Thus

$$\binom{r}{k}p^{r-k_i k}\sum_{j=0}^{p^{e-1}-1} j^{r-k} = \binom{r}{k}p^{r-k_i k}\alpha_{r,k}p^{e-1} \equiv_{p^e} 0 . \tag{6}$$

Thus for $k=1,2,\ldots,r-1$, $\sum_{i=0}^{p-1}\sum_{j=0}^{p^{e-1}-1}\binom{r}{k}j^{r-k}p^{r-k_i k} \equiv_{p^e} 0$.

Case 2: The first term of the binomial expansion, $r > k = 0$. Here, again substituting (5) into (4),

$$\binom{r}{k}p^{r-k_i k}\sum_{j=0}^{p^{e-1}-1} j^{r-k} = p^r\sum_{j=0}^{p^{e-1}-1} j^{r} = \begin{cases} p^r(-p^{e-2}+\alpha_{r,0}p^{e-1}) & \text{if } r = p-1 \\ p^r\alpha_{r,0}p^{e-1} & \text{if } r = 0,1,\ldots,p-2 \end{cases} \tag{7}$$

Now in (7), $p^r\alpha_{r,0}p^{e-1} \equiv_{p^e} 0$, since $r \geq 1$. Also in (7), $p^r(-p^{e-2}+\alpha_{r,0}p^{e-1}) \equiv_{p^e} 0$ if $r = p-1 \geq 2$, that is $p \geq 3$. Thus (7) becomes

$$\binom{r}{k}p^{r-k_i k}\sum_{j=0}^{p^{e-1}-1} j^{r-k} \equiv_{p^e} \begin{cases} -p^{e-1} & \text{if } p = 2 \\ 0 & \text{otherwise} \end{cases} . \tag{8}$$

Thus for $k = 0$, $\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k \equiv_{p^e} 0$, unless $p = 2$. But, even if $p = 2$ the sum is zero mod $p^e$, since $\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k$ reduces, by use of (8), to $\sum_{i=0}^{p-1} -p^{e-1}$, which is $-p^{e-1} \cdot p \equiv_{p^e} 0$.

Case 3: The last term of the binomial expansion, $r = k > 0$.

Here, $\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k$ reduces to

$$\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} i^r = p^{e-1} \sum_{i=0}^{p-1} i^r = \begin{cases} p^{e-1}(p-1+\alpha_{r,r}p) & \text{if } r = p-1 \\ p^{e-1}\alpha_{r,r}p & \text{if } r=0,1,\ldots,p-2 \end{cases}, \quad (9)$$

where Corollary 2 is used in obtaining the last equality. This reduces further to

$$\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} i^r \equiv_{p^e} \begin{cases} -p^{e-1} & \text{if } r = p-1 \\ 0 & \text{if } r=0,1,2,\ldots,p-2 \end{cases}. \quad (10)$$

Summing $\sum_{i=0}^{p-1} \sum_{j=0}^{p^{e-1}-1} \binom{r}{k} j^{r-k} p^{r-k} i^k$ over all $k$, and using the results of cases 1, 2, and 3, completes the proof of the lemma. ■

Lemma 8: If $p^e \mid N$, $p^{e+1} \nmid N$ and $e \geq 1$ then $p^e \nmid \sum_{i=0}^{N-1} i^{p-1}$.

Proof: $p^e \mid \sum_{i=0}^{N-1} i^{p-1}$ if and only if $\sum_{i=0}^{N-1} i^{p-1} \equiv_{p^e} 0$. Now

$\sum_{i=0}^{N-1} i^{p-1} \equiv_{p^e} \frac{N}{p^e} \sum_{i=0}^{p^e-1} i^{p-1}$. $p^{e+1} \nmid N$ implies $p \nmid \frac{N}{p^e}$ and, by Lemma 7,

$\sum_{i=0}^{p^e-1} i^{p-1} \equiv_{p^e} -p^{e-1}$. Thus $\frac{N}{p^e} \sum_{i=0}^{p^e-1} i^{p-1} \not\equiv_{p^e} 0$. ■

It is now possible to prove Theorem 10.

Proof of Theorem 10: As was pointed out earlier, all that there remains to prove is the first of the two statements found at the beginning of this section. To this end, suppose a collection of $[x,y]_p$-lines, p a prime factor of N, is given, $C = \{[x_i,y_i]_p\text{-lines} \mid i=1,2,\ldots,I\}$, so that there is no valid periodic skewing scheme for this collection, using p memory modules. The assumption of the existence of a valid periodic skewing scheme, using N memory modules, for the collection of $[x,y]_N$-lines, $C' = \{[x_i,y_i]_N\text{-lines} \mid i=1,2,\ldots,I\}$, will lead to a contradiction.

Suppose a valid periodic skewing scheme exists for C' using N memory modules. Call it $\varphi$. Then exactly N points on the torus must be mapped by the skewing scheme into memory module zero, one element from each of the N disjoint instances of the $[x_1,y_1]_N$-line. Let this set of N points be $\{(u_j,v_j) \mid j=0,1,\ldots,N-1\}$. Because $\varphi$ is assumed to be a valid periodic skewing scheme for C', using N memory modules, for any $[x_i,y_i]_N$-line in the collection each of the $(u_j,v_j)$ is a component of a different instance of the $[x_i,y_i]_N$-line. By Lemma 2, $\{x_iu_j - y_iv_j \bmod N \mid j=0,1,\ldots,N-1\} = \{0,1,2,\ldots,N-1\}$, for $i=1,2,\ldots,I$.

Thus, by Lemma 8, $p^e \nmid \sum_{j=0}^{N-1} ((x_iu_j - y_iv_j) \bmod N)^{p-1}$, for $i=1,2,\ldots,I$, where e is chosen so $p^e \mid N$, $p^{e+1} \nmid N$ and $e \geq 1$, this last since $p \mid N$ by

assumption. Note that $\sum_{j=0}^{N-1} ((x_i u_j - y_i v_j) \bmod N)^{p-1}$ does not depend on the choice of i. This sum will be denoted by $\Xi$.

Since there is assumed to be no valid periodic skewing scheme for C using p memory modules, by Lemma 4 either $(x_i, y_i, p) \neq 1$ for some i, or $I \geq p+1$ and there exist non-zero $g_i$ and a permutation, $\sigma$, for which $(x_{\sigma(1)}, y_{\sigma(1)}) \equiv_p (0, g_1)$, ..., $(x_{\sigma(p)}, y_{\sigma(p)}) \equiv_p (g_p(p-1), g_p)$, and $(x_{\sigma(p+1)}, y_{\sigma(p+1)}) \equiv_p (g_{p+1}, 0)$. Now if $(x_i, y_i, p) \neq 1$ for some i, then, since $p|N$, $(x_i, y_i, N) \neq 1$ also. This contradicts the assumed validity of $\varphi$. Therefore, I must be greater than or equal to p+1 and there must exist non-zero $g_i$ and a permutation, $\sigma$, with the required properties. Without loss of generality, assume $\sigma$ is the identity permutation.

Consider the system

$$\begin{bmatrix} y_1^{p-1} & y_2^{p-1} & \cdots & y_p^{p-1} \\ y_1^{p-2}x_1 & y_2^{p-2}x_2 & & y_p^{p-2}x_p \\ \vdots & \vdots & \cdots & \vdots \\ x_1^{p-1} & x_2^{p-1} & & x_p^{p-1} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_p \end{bmatrix} = \Delta \begin{bmatrix} y_{p+1}^{p-1} \\ y_{p+1}^{p-2}x_{p+1} \\ \vdots \\ x_{p+1}^{p-1} \end{bmatrix} \qquad (11)$$

where the matrix is called M, $\Delta = \det(M)$, and $R = [y_{p+1}^{p-1} \cdots x_{p+1}^{p-1}]^T$, and $\Gamma = [\gamma_1 \ \gamma_2 \cdots \gamma_p]^T$ are unknowns to be determined.

An important question is: Does this system have a solution, and, if so, is it unique? The answer to both parts of this question will be yes if $\Delta \neq 0$. In order to prove $\Delta \neq 0$ and to obtain some information about the form of the $\gamma_i$, the system in (11) can be converted into a similar system in the field of residue classes mod p.

When the elements of M are replaced by their values in the field of residue classes mod p, the resulting matrix is the $M_p$ of Lemma 6. To see this observe that $M_{i,j} = y_j^{p-i} x_j^{i-1} \equiv_p g_j^{p-i}(g_j(j-1))^{i-1} \equiv_p g_j^{p-1}(j-1)^{i-1} \equiv_p (j-1)^{i-1} = M_{p_{i,j}}$, where $(x_j, y_j) \equiv_p (g_j(j-1), g_j)$ was used, (recall $\sigma$ was assumed to be the identity permutation), and $g_j^{p-1} \equiv_p 1$ by Fermat's theorem. This observation justifies the choice of the name $M_p$ in Lemma 6.

Also notice that $\Delta \bmod p = \det(M_p)$ where the determinant of $M_p$ is calculated in the field of residue classes mod p. This is nothing more than observing that the mod operator and the det operator commute. Because $\Delta \bmod p = \det(M_p)$, $\Delta_p$ is a reasonable notation to use for either of these quantities. In a manner similar to that used in the proof that M is converted to $M_p$ by the mod operator, R is converted to $R_p = [0 \;\; 0 \;\; \ldots \;\; 0 \;\; 1]^T$.

By Corollary 1 of Lemma 6, $\Delta_p \not\equiv_p 0$, so $\Delta \neq 0$. Thus the system $M\Gamma = \Delta R$ and the system $M_p X = \Delta_p R_p$ have unique solutions. The reader is cautioned that despite the fact that both systems have unique solutions, it is not obvious that the mod operator applied to the $\gamma_i$ converts $\Gamma$ to X. The reason for this is that the $\gamma_i$ might not be integers, i.e., if the $\gamma_i$ are only rational it makes no sense to consider $\gamma_i$ mod p. If, however, all the $\gamma_i$ are integers then X will in fact be $\Gamma_p$, the column vector obtained by replacing $\gamma_i$ with $\gamma_i$ mod p.

It is possible to show, however, that $\gamma_i$ is an integer by solving $M\Gamma = \Delta R$ by means of Cramer's rule. When using Cramer's rule, $\gamma_i$ is calculated by replacing column i of M by $\Delta R$, getting a new matrix, $\hat{M}_i$, and then $\gamma_i = \dfrac{\det(\hat{M}_i)}{\det(M)}$.

By using common rules for manipulating determinants [8], $\gamma_i = \frac{\Delta \det(\hat{M}_i')}{\Delta} = \det(\hat{M}_i')$, where $\hat{M}_i'$ is the matrix formed by replacing column i of M by R. Since every element of $\hat{M}_i'$ is an integer, $\det(\hat{M}_i)$ is also an integer. Thus $\Gamma$ consists solely of integers, and $X = \Gamma_p$.

In order to complete the argument, by arriving at the contradiction mentioned at the beginning of the proof, it is convenient to determine the form of $\Gamma$. This can be done by first determining the form of $\Gamma_p$. This is easily done directly. In the proof of Corollary 1 of Lemma 6, it was shown that $M_p^{-1} = (p-1)M_p'$. As noted earlier

$$R_p = [0 \quad 0 \ \ldots \ 0 \quad 1]^T, \text{ so } \Gamma_p = (p-1)M_p'\Delta_p R_p = (p-1)\Delta_p \begin{bmatrix} 1 \\ 1^{-(p-1)} \\ 2^{-(p-1)} \\ \vdots \\ (p-1)^{-(p-1)} \end{bmatrix},$$

since $\begin{bmatrix} 1 \\ 1^{-(p-1)} \\ 2^{-(p-1)} \\ \vdots \\ (p-1)^{-(p-1)} \end{bmatrix}$ is the last column of $M_p'$. But by Fermat's theorem

$$\begin{bmatrix} 1 \\ 1^{-(p-1)} \\ 2^{-(p-1)} \\ \vdots \\ (p-1)^{-(p-1)} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \text{ so } \Gamma_p = \begin{bmatrix} \Delta_p(p-1) \\ \Delta_p(p-1) \\ \Delta_p(p-1) \\ \vdots \\ \Delta_p(p-1) \end{bmatrix}.$$ Renaming

$\Delta_p(p-1) = c$, it follows that $\Gamma = \begin{bmatrix} c+\delta_1 p \\ c+\delta_2 p \\ \vdots \\ c+\delta_p p \end{bmatrix}$, where the $\delta_i$ may all be distinct.

Finally consider the sum

$$(c+p\delta_1)\sum_{j=0}^{N-1}(x_1u_j-y_1v_j)^{p-1}+\cdots+(c+p\delta_p)\sum_{j=0}^{N-1}(x_pu_j-y_pv_j)^{p-1} \equiv_N$$

$$(c+p\delta_1)\Xi + (c+p\delta_2)\Xi + \cdots + (c+p\delta_p)\Xi \equiv_N pc\Xi + p\Xi(\delta_1+\delta_2+\cdots+\delta_p) \equiv_N$$

$p\Xi\nabla$, where $\nabla = \delta_1+\delta_2+\cdots+\delta_p+c$.

Now expand the sum another way.

$$(c+p\delta_1)\sum_{j=0}^{N-1}(x_1u_j-y_1v_j)^{p-1}+\cdots+(c+p\delta_p)\sum_{j=0}^{N-1}(x_pu_j-y_pv_j)^{p-1} =$$

$$(c+p\delta_1)x_1^{p-1}\sum_{j=0}^{N-1}u_j^{p-1} + (c+p\delta_1)(p-1)(-1)x_1^{p-2}y_1\sum_{j=0}^{N-1}u_j^{p-2}v_j+\cdots+$$

$$(c+p\delta_1)(-1)^{p-1}y_1^{p-1}\sum_{j=0}^{N-1}v_j^{p-1}$$

$$+ (c+p\delta_2)x_2^{p-1}\sum_{j=0}^{N-1}u_j^{p-1} + (c+p\delta_2)(p-1)(-1)x_2^{p-2}y_2\sum_{j=0}^{N-1}u_j^{p-2}v_j+\cdots+$$

$$(c+p\delta_2)(-1)^{p-1}y_2^{p-1}\sum_{j=0}^{N-1}v_j^{p-1} \qquad (1$$

$$+$$

$$\vdots$$

$$+ (c+p\delta_p)x_p^{p-1}\sum_{j=0}^{N-1}u_j^{p-1} + (c+p\delta_p)(p-1)(-1)x_p^{p-2}y_p\sum_{j=0}^{N-1}u_j^{p-2}v_j+\cdots+$$

$$(c+p\delta_p)(-1)^{p-1}y_p^{p-1}\sum_{j=0}^{N-1}v_j^{p-1}.$$

If the terms in the sum of equation (12) are added together by columns, the result is

$$\left(\sum_{j=0}^{N-1} u_j^{\,p-1}\right)\left(\sum_{i=1}^{p} (c+p\delta_i)x_i^{\,p-1}\right)+(p-1)(-1)\left(\sum_{j=0}^{N-1} u_j^{\,p-2}v_j\right)\left(\sum_{i=1}^{p} (c+p\delta_i)\,x_i^{\,p-2}y_i\right)$$

$$+\cdots+(-1)^{p-1}\left(\sum_{j=0}^{N-1} v_j^{\,p-1}\right)\left(\sum_{i=1}^{p} (c+p\delta_i)y_i^{\,p-1}\right). \tag{13}$$

But by considering the system $M\Gamma = \Delta R$, and the value for $\Gamma$ derived above, $\sum_{i=1}^{p} (c+p\delta_i)x_i^{\,p-k}y_i^{\,k-1} = \Delta x_{p+1}^{\,p-k}y_{p+1}^{\,k-1}$, and when this is substituted into (13) the sum becomes

$$\left(\sum_{j=0}^{N-1} u_j^{\,p-1}\right)\Delta x_{p+1}^{\,p-1}+(p-1)(-1)\left(\sum_{j=0}^{N-1} u_j^{\,p-2}v_j\right)\Delta x_{p+1}^{\,p-2}y_{p+1}$$

$$+\cdots+(-1)^{p-1}\left(\sum_{j=0}^{N-1} v_j^{\,p-1}\right)\Delta y_{p+1}^{\,p-1} = \Delta\sum_{j=0}^{N-1}(x_{p+1}u_j-y_{p+1}v_j)^{p-1} \equiv_N \Delta\Xi$$

Combining this with the form of the sum derived earlier, $p\,\Xi\nabla \equiv_N \Delta\Xi$ or $\Xi(p\nabla-\Delta) \equiv_N 0$. Now e was chosen earlier in the proof so that $p^e|N$, but $p^e \nmid \Xi$. However, if $\Xi(p\nabla-\Delta) \equiv_N 0$ then $p^e|\Xi(p\nabla-\Delta)$. Thus p must divide $p\nabla-\Delta$. Since $p|p\nabla$, this implies $p|\Delta$. But $p \nmid \Delta$, since $\Delta_p \neq_p 0$. This is the desired contradiction, and the theorem is proven. ■

## 3.5 Further Results and Examples

In Sections 3.1 through 3.4 it was shown that given N memory modules and a collection of $[x,y]_N$-lines, restricting consideration to linear skewing schemes suffices to determine the existence of valid periodic skewing schemes for the collection. It is natural to

ask whether this result can be extended further, so that by consideration of only linear skewing schemes, the question of the existence of an arbitrary skewing scheme, valid in the plane, can be settled. A limited answer is given by

Theorem 11: Given N, the number of memory modules, and a collection of $[x,y]_N$-lines, $\{[x_i,y_i]_N$-lines$\mid i=1,2,\ldots,I\}$, if there exists two sequences of integers, $(a_1,a_2,\ldots,a_I)$ and $(b_1,b_2,\ldots,b_I)$, such that $\sum_{i=1}^{I} (a_i x_i, a_i y_i) = (0,1)$ and $\sum_{i=1}^{I} (b_i x_i, b_i y_i) = (1,0)$, then if there does not exist a valid linear skewing scheme for this collection, there is no valid skewing scheme for this collection.

Proof: The proof is quite simple. The conditions of this theorem and the result of Theorem 10, imply there is no valid periodic skewing scheme for this collection of $[x,y]_N$-lines. The conditions on the sequences of $a_i$'s and $b_i$'s will be seen to imply that if there are any valid skewing schemes, then they are periodic. This will establish the theorem.

Notice that for any valid skewing scheme, $\varphi$, $\varphi(i,j) = \varphi(i+Ny_k, j+Nx_k)$, for $k=1,2,\ldots,I$ and any i and j. This is so, because $\varphi(i,j)$, $\varphi(i+y_k, j+x_k), \ldots, \varphi(i+(N-1)y_k, j+(N-1)x_k)$ must all be distinct, since $((i,j),(i+y_k,j+x_k),\ldots,(i+(N-1)y_k,j+(N-1)x_k))$ is an instance of the $[x_k,y_k]_N$-line. Similarly, $\varphi(i+y_k,j+x_k)$, $\varphi(i+2y_k,j+2x_k)$, ..., $\varphi(i+Ny_k,j+Nx_k)$ must all be distinct, since $((i+y_k,j+x_k), (i+2y_k,j+2x_k), \ldots, (i+Ny_k,j+Nx_k))$ is also an instance of the $[x_k,y_k]_N$-line. Since these two instances have N-1 ordered pairs in common, the pigeonhole principle requires that

$\varphi(i,j) = \varphi(i+Ny_k, j+Nx_k)$. From this it is clear that $\varphi(i,j) = \varphi(i+\ell Ny_k, j+\ell Nx_k)$, for $\ell = \ldots,-1,0,1,\ldots$ and any i and j.

Since k was arbitrary, it follows that for any (c,d),

$$\varphi(c,d) = \varphi(c + \sum_{i=1}^{I} a_i Ny_i,\ d + \sum_{i=1}^{I} a_i Nx_i) = \varphi(c + N \sum_{i=1}^{I} a_i y_i, d + N \sum_{i=1}^{I} a_i x_i) =$$

$\varphi(c+N,d)$. Similarly, by using the sequence of $b_i$'s, $\varphi(c,d) = \varphi(c,d+N)$. Since (c,d) was arbitrary, $\varphi(c,d) = \varphi(c+N,d) = \varphi(c,d+N)$ establishes that $\varphi$ must be a periodic skewing scheme. ■

This condition, restrictive though it may be, is sufficient to resolve the most important practical case: $\{[1,0]_N$-line, $[0,1]_N$-line, $[1,1]_N$-line, $[1,-1]_N$-line$\}$. The sequences (0,1,0,0) and (1,0,0,0) suffice, clearly. Thus for this important case, considered by Budnik and Kuck [3] and Lawrie [10], if there does not exist a linear skewing scheme using N memory modules, and there does not when N is a power of two, then there is no valid skewing scheme of any type whatsoever.

It is easy to allow oneself to be misled by the conclusion of Theorem 10. When given a collection of $[x,y]_N$-lines, and deciding on a skewing scheme using N memory modules, there may be advantages to choosing a non-linear, but still periodic, skewing scheme. Some periodic skewing schemes can be so simple that they take very little hardware to perform address computation and to align the data with the correct processor, even less hardware than required by linear skewing schemes. One such periodic skewing scheme has been used in the construction of an actual machine, the STARAN [1]. Abstracting from the exact details of the STARAN design, the programmer views memory as consisting of a $2^n \times 2^n$ array. In the language of the designers of the STARAN, the programmer views the memory as having $2^n$ words of $2^n$ bits, and the programmer can indicate

he wants to fetch all the bits of one word, or a bit-slice, the $j^{th}$ bit of each word (Figure 22).† In the terminology used here, this is equivalent to fetching arbitrary instances of the $[1,0]_{2^n}$-line and the $[0,1]_{2^n}$-line from an array of data elements, using $2^n$ memory modules to store the data, and using a periodic skewing scheme.

The skewing scheme employed is $\varphi(i,j) = i \bmod 2^n \oplus j \bmod 2^n$, where $i \bmod 2^n$ and $j \bmod 2^n$ are expressed in binary notation and $\oplus$ is exclusive-or. This periodic skewing scheme is explicitly calculated for an $8 \times 8$ array in Figure 23. Unlike the other machine designs discussed earlier, the responsibility of deciding on a skewing scheme does not rest with the programmer or compiler, but is built directly into the hardware. Indeed the user of the STARAN need not even know that a skewing scheme is employed; by appropriately setting the global address register, G, and the access mode register, M, either the correct word or bit-slice will be made to appear at the processing elements. Additional ways of setting M allow some, but not all, instances of other generalized lines (but not $[x,y]_{2^n}$-lines) to be fetched without memory conflict.

The reason that this skewing scheme is of practical importance is that the address computations needed to fetch instances of the $[1,0]_{2^n}$-line and the $[0,1]_{2^n}$-line can be done using only n exclusive-or gates. This compares to $2^n$ adders that would be needed if a linear skewing scheme was employed. Additionally, an exclusive-or can be performed in less time than

---

†What has been called memory here is actually the memory of a single array module in the STARAN. Each array module has a memory consisting of a $256 \times 256$ array of bits. This memory physically consists of 256 independent memory modules, each with 256 one-bit words. By the skewing scheme described in the body of the text all the bits of any word and all the bits of any bit-slice lie in different memory modules, and can be fetched in one memory cycle. Readers interested in exact implementation details and the terminology used by the STARAN designers should consult [1].

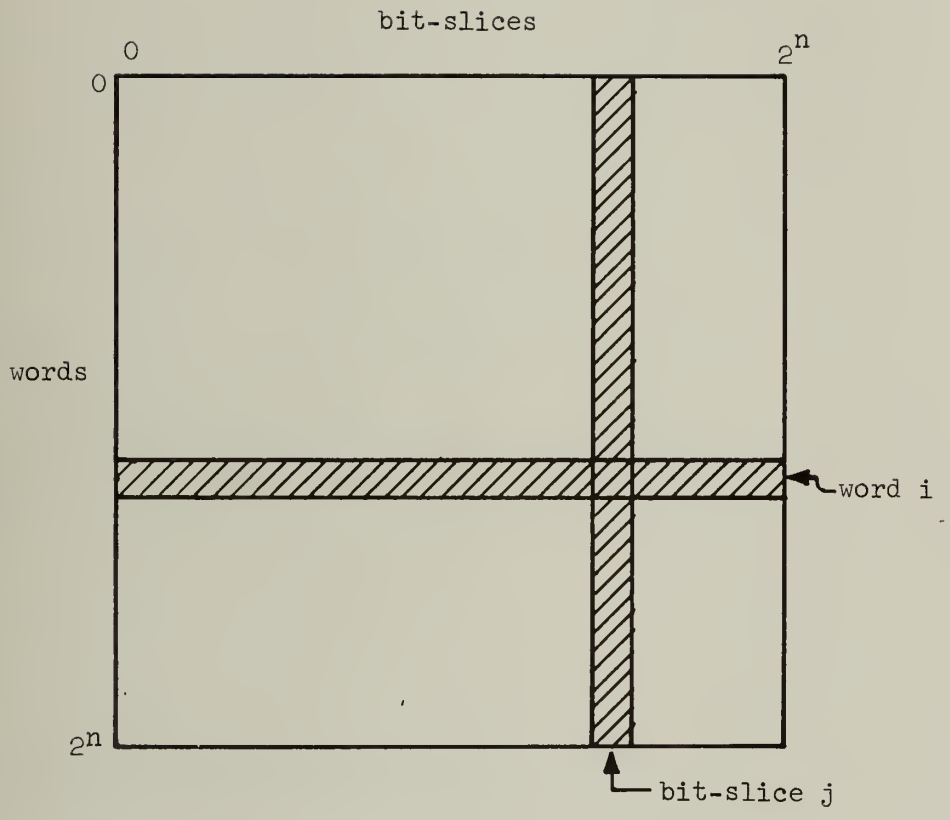



Figure 22: Programmer's View of STARAN Memory.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 |
| 2 | 3 | 0 | 1 | 6 | 7 | 4 | 5 |
| 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 |
| 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 |
| 5 | 4 | 7 | 6 | 1 | 0 | 3 | 2 |
| 6 | 7 | 4 | 5 | 2 | 3 | 0 | 1 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

Figure 23: The Periodic Skewing Scheme Used in the STARAN Computer.

an addition, since there is no carry propagation. The memory-processor connection network required in the STARAN is of some order of complexity as Lawrie's Ω-network. The reader interested in the details of address computation, a proof of the validity of the skewing scheme, and the details of the memory-processor connection network should consult Batcher [1].

This example was presented to show that non-linear periodic skewing schemes can be important in actual practice, even when there are valid linear skewing schemes. In comparing STARAN to the more general machine modeled in Figure 3, it should be noted that in STARAN the generalized lines that the programmer can access conveniently were fixed at the time of design. For these generalized lines the programmer need not concern himself with skewing schemes, as the hardware handles the data storage and unscrambling automatically. In the more general computer, modeled in Figure 3, the determination of an appropriate skewing scheme is left to the programmer, who may be restricted in his choices by the nature of the memory-processor connection network. While this may be more work for the user, it allows greater flexibility than is available in the STARAN. Some authors [11] have discussed leaving the choice of skewing scheme and/or the address computation to the compiler, thus freeing the programmer from this bookkeeping.

# 4. UNRESOLVED PROBLEMS AND DIRECTIONS OF FURTHER RESEARCH

## 4.1 The Effectiveness of Linear and Periodic Skewing Schemes

One question that has occurred throughout this thesis is: Given a collection of generalized lines, when can the search for a valid skewing scheme for this collection safely be restricted to certain subclasses of skewing schemes, that is, when does a valid skewing scheme from a class of skewing schemes, imply a valid skewing scheme from a subclass of this class of skewing schemes? In Chapter 1, it was shown that for any collection of generalized lines, attention could safely be restricted from skewing schemes valid on the quarter plane to skewing schemes valid on the entire plane. Similarly, Theorem 10 shows that for collections of $[x,y]_N$-lines, attention can safely be restricted from periodic skewing schemes, using N memory modules, to linear skewing schemes, using N memory modules. In general, for an arbitrary collection of generalized lines, the question of when attention can be restricted from arbitrary skewing schemes defined on the plane to periodic skewing schemes, and from periodic skewing schemes to linear skewing schemes is unresolved. In this section some conjectures, partial results, and interesting examples are presented for a class of generalized lines called polyominoes [5,6].

> Definition 11: A polyomino is a generalized line in which given any two components, $(x_\nu,y_\nu)$ and $(x_\tau,y_\tau)$, there exists a path $(x_\nu,y_\nu) = (x_{i_1},y_{i_1}),(x_{i_2},y_{i_2}),\ldots,$

$(x_{i_r}, y_{i_r}) = (x_\tau, y_\tau)$ such that $(x_{i_j}, y_{i_j})$ is a component of the generalized line, for $j=1,2,\ldots,r$ and either $x_{i_{j+1}} = x_{i_j} \pm 1$ and $y_{i_{j+1}} = y_{i_j}$ or $x_{i_{j+1}} = x_{i_j}$ and $y_{i_{j+1}} = y_{i_j} \pm 1$, for $j=1,2,\ldots,r-1$.

In the geometric realization of a generalized line by unit squares, a polyomino is a generalized line that is connected.† Except for the disconnected shape of Figure 20, all the generalized lines used as examples in Chapter 2 are polyominoes. There is an impressive amount of literature, and many unsolved problems concerning polyominoes. A general source is [6].

Conjecture: Given N memory modules, and a polyomino of length N, then if there is a valid skewing scheme for the polyomino, there is also a valid periodic skewing scheme for the polyomino.

This conjecture is supported by consideration of a construction, illustrated initially by example. Consider the generalized line whose geometric realization is . Theorem 5 proves that the problem of finding valid skewing schemes, is equivalent to determining tesselations of the plane. With the objective of analyzing possible tesselations, lay down an instance of this generalized line (see Figure 24). Without loss of

†In the literature, polyominoes are usually defined to be the geometric realizations of the class of generalized lines described by Definition 11. Additionally, unlike here, in most problems concerning polyominoes, rotations and reflections of a polyomino are permitted and are not regarded as generating different polyominoes. Additionally, a comment should be made about connectedness. Here, connected means connected by more than a corner. This kind of connectedness has been called rook-wise connected, because of the permissible motions of the chess piece by the same name.

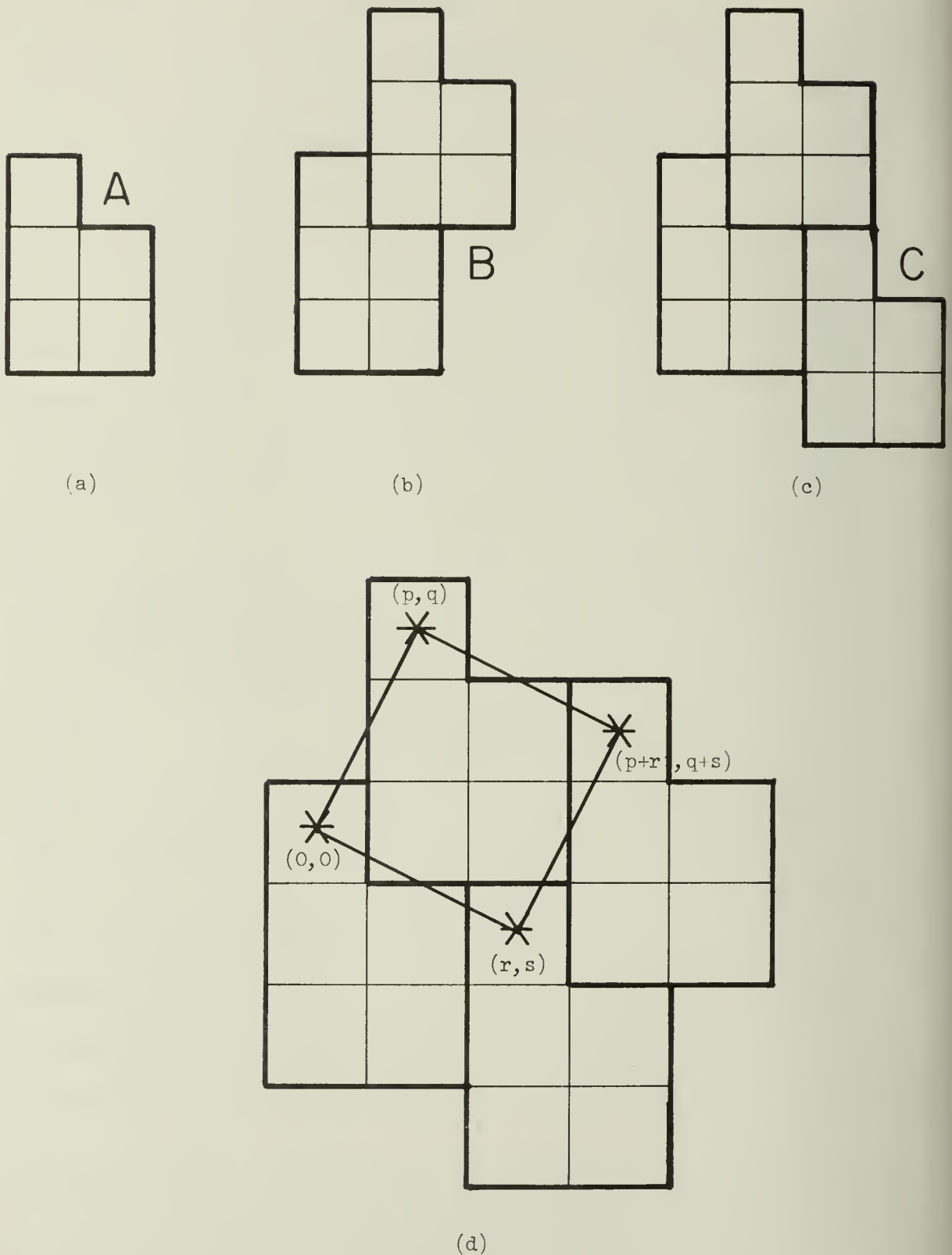
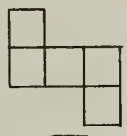
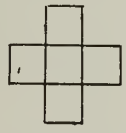
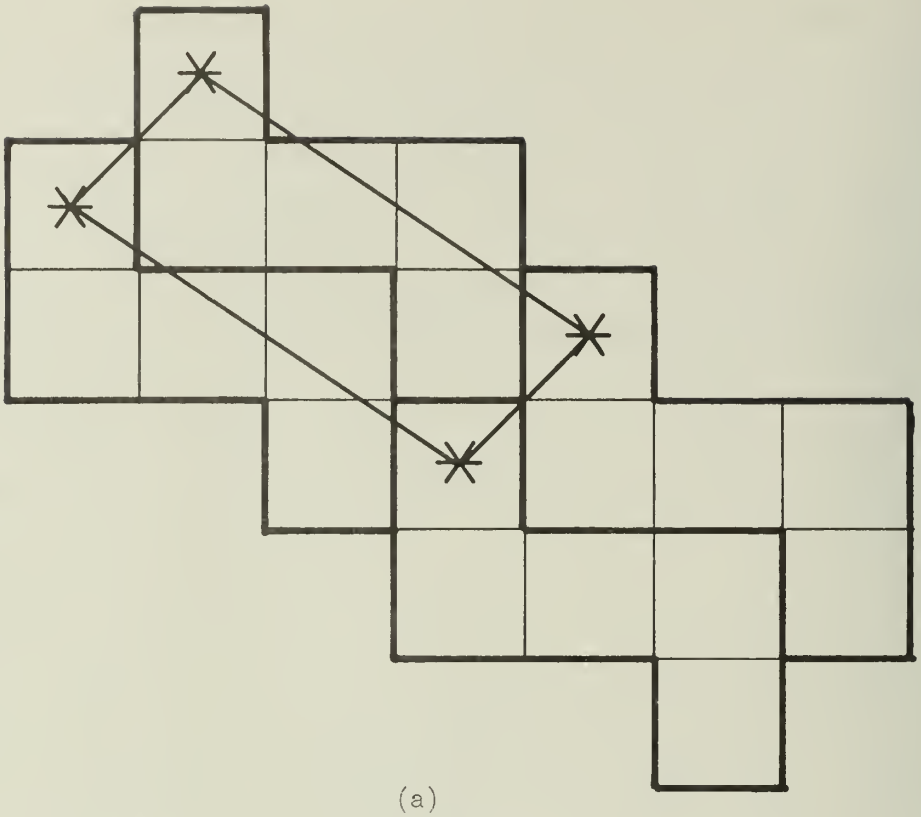



Figure 24: Positioning Four Instances of the Generalized Line ((0,0),(1,0),(1,1),(2,0),(2,1)), so their Designated Elements Form a Parallelogram.

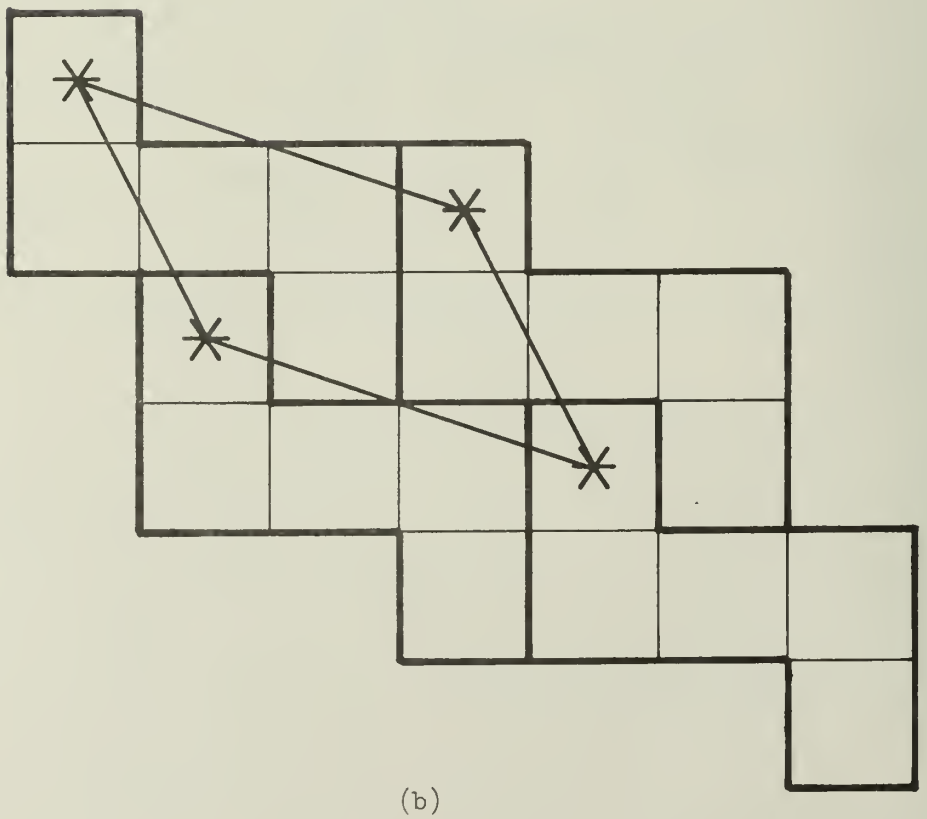
generality the designated element can be assumed to be at (0,0). Now, there is only one way a second instance can be positioned so that the square labeled A is covered without overlapping of the instances (Figure 24(b)). Again there is only one way to position a third instance so the square labeled B is covered without any overlapping of the instances (Figure 24(c)). Finally, there is only one way for a fourth instance to be positioned so that the square labeled C is covered and, again, there is no overlapping of the instances (Figure 24(d)). Notice that the designated elements form the vertices of a parallelogram which contains "no holes," and has area N, the length of the polyomino.† Now, by replication of the parallelogram, it is clear that a tesselation of the plane results. The tesselation that results is very orderly. Sometimes, particularly when the polyomino has a high degree of symmetry, the construction, informally presented above, can yield more than one parallelogram. This is illustrated, in Figure 25, by the generalized line, whose geometric realization is  -shaped. The generalized line, whose geometric realization is  -shaped, exhibits the same phenomenon.

When four instances of a polyomino of length N can be laid down so their designated elements form the four vertices of a parallelogram of area N and which contains no holes, then the tesselation of the plane, produced by replicating the parallelogram, induces a periodic skewing scheme. The proof of this statement is reminiscent of the proof of Theorem 11. If the vertices of the parallelogram are labeled as in

†In this example, N = 5.



(a)



(b)

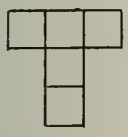
Figure 25: Alternate Positionings of Instances for the Generalized Line ((0,0),(1,0),(1,1),(1,2),(2,2)).

Figure 24(d), then letting $\varphi$ be the skewing scheme induced by the tesselation, it is clear that

$$\varphi(c,d) = \varphi(c+p,d+q) = \varphi(c+r,d+s), \tag{14}$$

for any $(c,d)$. Now $|ps-qr| = N$ since $|ps-qr|$ is the area of the parallelogram, and thus $(c,d)+(-rp,-rq)+(pr,ps) = (c,d+N)$ (or $(c,d-N)$, depending on the sign of $ps-qr$). Combining this with (14) implies $\varphi(c,d) = \varphi(c,d+N)$. Similarly, $(c,d)+(sp,sq)+(-qr,-qs) = (c+N,d)$ (or $(c-N,d)$), and, thus, $\varphi(c,d) = \varphi(c+N,d)$. Since $(c,d)$ was arbitrary, $\varphi$ is periodic.

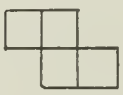
The obstacle standing in the way of a proof of the conjecture is the inability to prove that if a polyomino tesselates the plane, which implies the existence of a valid skewing scheme, then four instances of the polyomino can be positioned so that the designated elements form the vertices of a parallelogram of area N. This has been checked by hand for all polyominoes through N = 7 and no exceptions have been found. It is reasonable to believe that this is in fact so, since, for most polyominoes there is usually only one way a second instance can be positioned so that some carefully chosen square is covered, and at the same time the instances do not overlap. This style of argument was used to show that the



cannot tesselate the plane (Figure 12), as well as to construct the unique tesselation of the plane (except for rigid shifting) by the

, (Figures 14 and 24). Polyominoes with some symmetry, however, frequently produce several distinct tesselations.

Even though it seems reasonable to believe that if a polyomino tesselates the plane, the construction of a parallelogram of area N will always be possible, there is some evidence to the contrary. Some authors have considered tesselating the plane, while allowing simultaneous use of several different polyominoes.† Examples have been reported of collections of polyominoes from which a tesselation of the plane can be constructed, but from which no periodic tesselation (with any period whatsoever) can be constructed [5].

Because the example presented below indicates that there may be some unrecognized subtleties, to close the discussion of this conjecture, an alternate approach to its proof, known to be inadequate, will be discussed. Suppose a polyomino of length N is given, and $\psi$ is a valid skewing scheme for this polyomino. Then, define $\varphi(i,j) = \psi(i \bmod N, j \bmod N)$. $\varphi$ is periodic, clearly. The objective is to show that $\varphi$ is valid. This approach is motivated by consideration of the $[1,0]_N$-line. Clearly, any valid skewing scheme, $\psi$, has the property $\psi(c,d) = \psi(c,d+N)$ for any $(c,d)$. However, it is easy to construct $\psi$ so that $\psi(c,d) \neq \psi(c+N,d)$, i.e. $\psi$ is non-periodic in the vertical direction. Note, however, that $\varphi$, defined by $\varphi(i,j) = \psi(i \bmod N, j \bmod N)$ is both periodic and valid for the $[1,0]_N$-line, provided $\psi$ is valid. This technique also works on generalized lines whose geometric realization is an $N_1 \times N_2$ rectangle, where $N_1 N_2 = N$. However, consideration of the generalized line $L = ((0,0),(0,1),(1,1),(1,2))$, whose geometric is  , shows that such an approach to the proof of

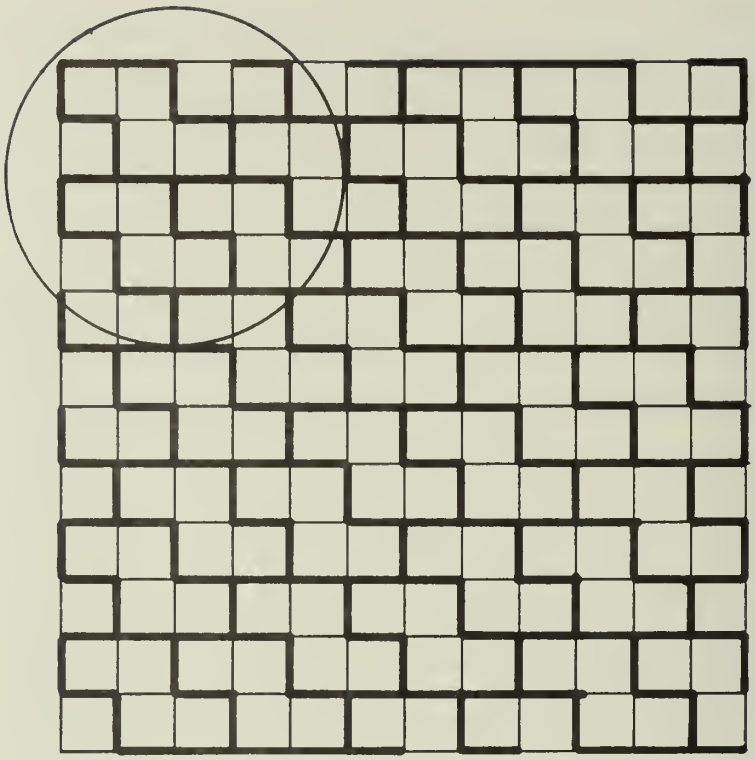
†These tesselations should be carefully distinguished from those used in Theorem 6. There, several tesselations were constructed using different generalized lines, but each tesselation used instances of only one type. In the case mentioned here, several different polyominoes were used to construct a single tesselation.

the conjecture is inadequate. Figure 26 indicates a tesselation, which induces a valid skewing scheme, $\psi$, for which $\varphi$, defined by $\varphi(i,j) = \psi(i \bmod N, j \bmod N)$ is not a valid skewing scheme. There are, however, valid periodic skewing schemes for L.

If this conjecture is true, then given N memory modules and a polyomino of length N, when looking for a valid skewing scheme attention can safely be restricted to periodic skewing schemes. A good question to ask is: Can attention safely be restricted to linear skewing schemes? While the answer to this question is "no," in general, if N is prime and if the existence of a tesselation by a polyomino implies that the construction described earlier results in a parallelogram of area N, then the answer is "yes." Using the notation of Figure 24(d), for $\varphi(i,j) = ai+bj \bmod N$, define a and b by:

$$\text{if } (p,q) \not\equiv (0,0) \text{ then } a = -q,\ b = p$$
$$\text{else } a = 1,\ b = 1\ .$$

To see that $\varphi$ is valid, observe first that for the exceptional case $(p,q) \equiv (0,0)$, either $(p,q) = (\pm N, 0)$ or $(p,q) = (0, \pm N)$. This is true because the polyomino is connected and has only N components. Now $(p,q) = (\pm N, 0)$ or $(0, \pm N)$ implies that the polyomino is a $[0,1]_N$-line or a $[1,0]_N$-line, and then $\varphi(i,j) = i+j \bmod N$ is indeed valid. When $a = -q$ and $b = p$, that $\varphi$ is valid can be seen as follows. Note that $\varphi(p,q) = -qp+pq \bmod N = 0$, and $\varphi(r,s) = -qr+ps \bmod N = 0$. Thus adding multiples of $(p,q)$ and/or $(r,s)$ to a point does not effect the value of $\varphi$. Because of this, and the orderly way in which replications of the parallelogram tesselate the plane, if $\varphi$ maps all the components of one instance to distinct memory modules, then $\varphi$ is valid. However, by adding



Enlargement of the $4 \times 4$ square, showing the skewing scheme induced by the above tesselation

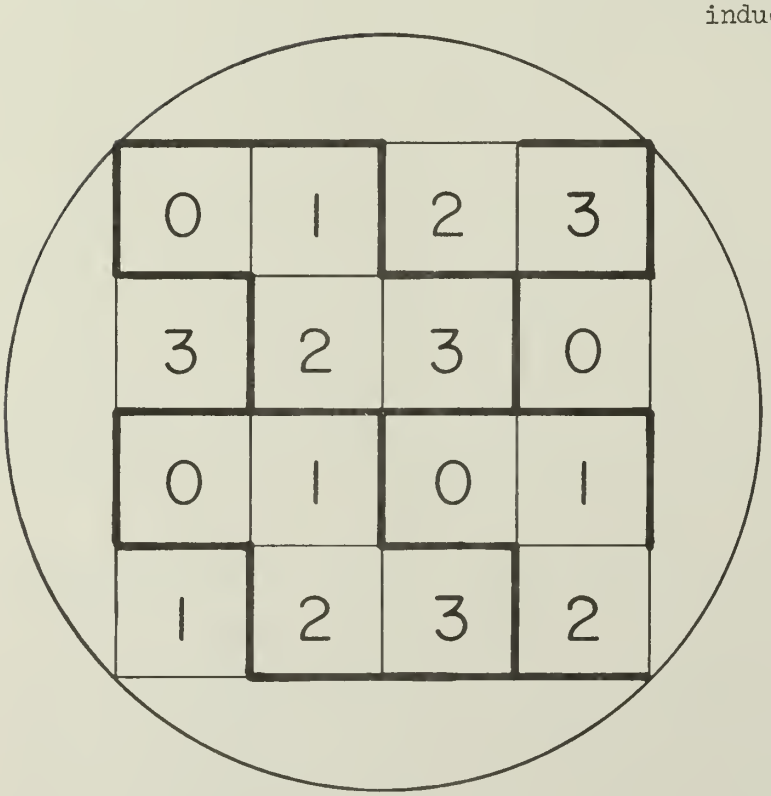



Figure 26: A Non-periodic Skewing Scheme, $\psi$, for which $\Phi(i,j) = \psi(i \bmod N, j \bmod N)$ is not Valid.

(p,q) and/or (r,s) to some components of the L(0,0) instance, where L is the polyomino, the condition on φ can be changed to: φ maps into distinct memory modules all ordered pairs (of integers) interior to the parallelogram or lying on the edges whose endpoints are (0,0) and (p,q), and (0,0) and (r,s), but not including (p,q) and (r,s). Figure 27 illustrates the situation. Now, φ will be the same for two ordered pairs of integers interior to or lying on the edges of the parallelogram if and only if the line segment connecting them is parallel to the line segment from (0,0) to (p,q), since only then will $|-qi_1+pj_1|$, the area of parallelogram I in Figure 28, equal $|-qi_2+pj_2|$, the area of parallelogram II in Figure 28. The existence of two ordered pairs of integers interior to or lying on the edges of the parallelogram, for which the line segment connecting them is parallel to the line segment connecting (0,0) to (p,q) implies the existence of an ordered pair of integers lying on the line segment connecting (0,0) to (p,q). But this can occur only if p and q have a common divisor other than one. Then this common divisor, which is less than N, divides $|ps-qr| = N$, which is impossible since N is prime. This contradiction implies φ is a valid linear skewing scheme.

When N is not a prime number, it is not always possible to restrict attention to linear skewing schemes, when determining if valid skewing schemes exist for a polyomino. Figure 29 illustrates that four instances of the generalized line L = ((0,0),(0,1),(0,2),(0,3),(1,0),(1,1),(1,2),(1,3),(2,0),(2,1),(3,0),(3,1)) can be positioned so that the designated elements form a parallelogram of area N (N = 12). Thus there is a valid periodic skewing scheme for L. There is no valid linear skewing scheme for L, as trying all the possibilities indicates. Notice that the

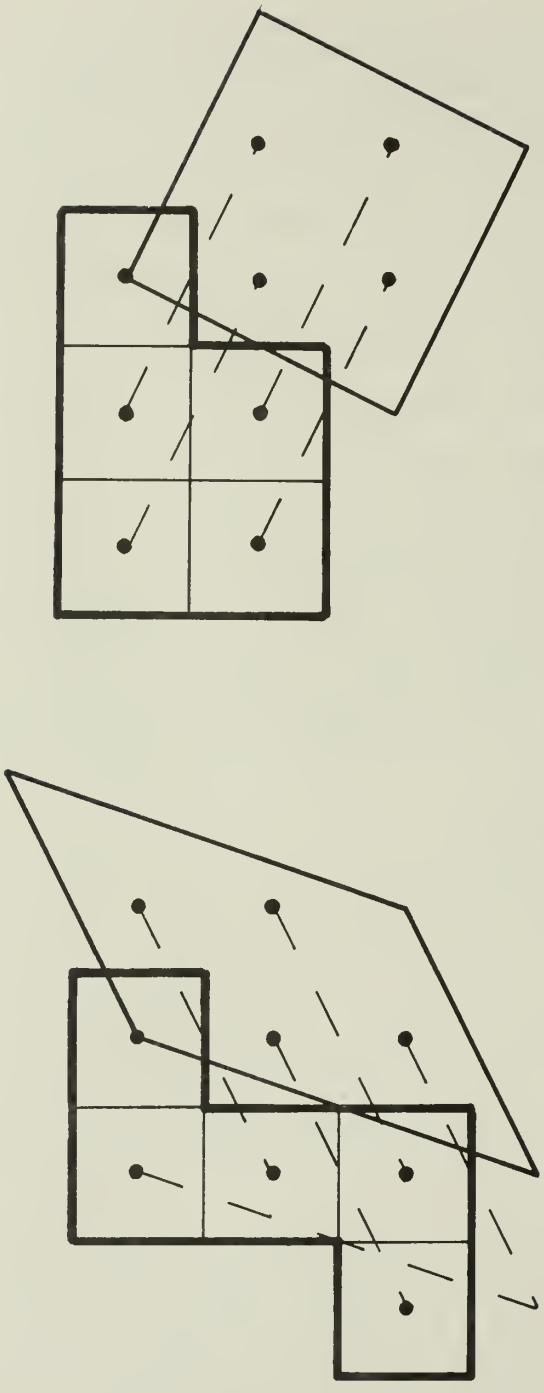


Figure 27: Examples of Translating by (p,q) and/or (r,s), so that all the Components of the Instance of the Generalized Line Lie Interior to the Parallelogram.

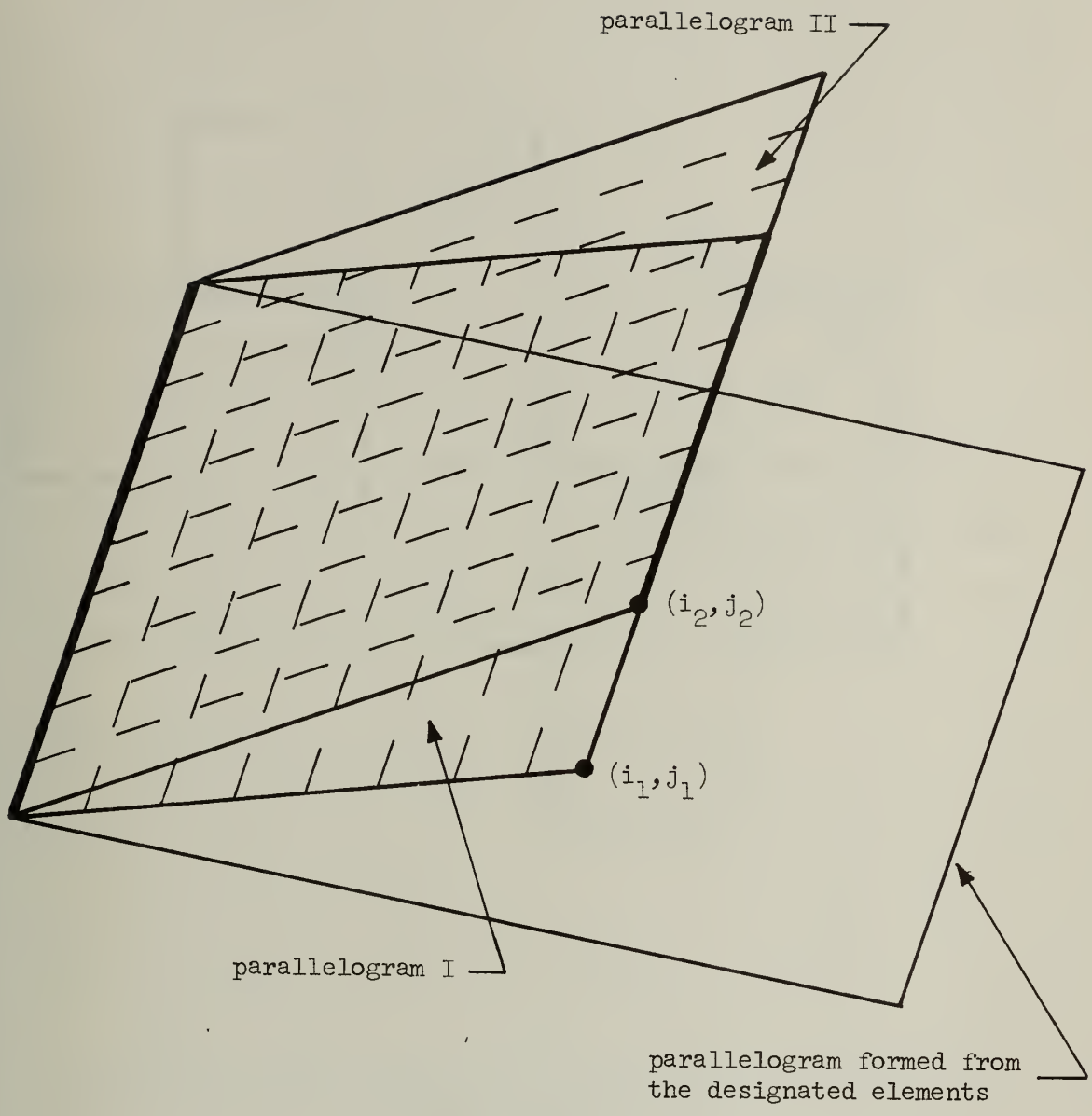



Figure 28: Components of an Instance of a Generalized Line, after Translation by (p,q) and/or (r,s), which are Stored in Same Memory Module.

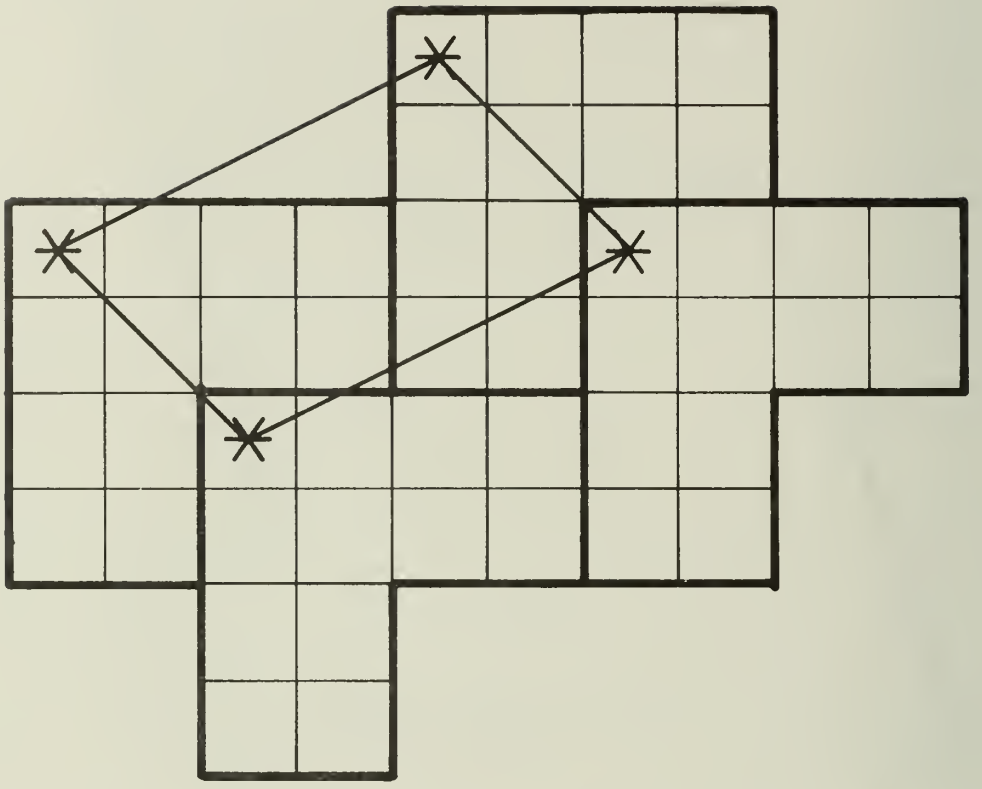


Figure 29: An Example of a Polyomino for which There is a Valid Periodic Skewing Scheme, but no Valid Linear Skewing Scheme.

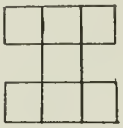
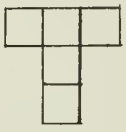
line segments connecting (0,0) and (p,q) and (0,0) and (r,s) pass through ordered pairs of integers. It is frequently the case that for N, a composite number, one of these line segments will not pass through any ordered pairs of integers. In such a case, as in the proof outlined above, $\varphi(i,j) = ai+bj \bmod N$, will be a valid linear skewing scheme, where $a = -q$ and $b = p$ or $a = -s$, $b = r$, depending on which line segment does not pass through an ordered pair of integers.

In summary, there is evidence to believe that in determining the existence of a valid skewing scheme for a polyomino, using the same number of memory modules as the length of the polyomino, only periodic skewing schemes need be considered, and if N is prime attention can be restricted further, to linear skewing schemes.

## 4.2 Questions Relating to Memory Utilization

Throughout much of this work attention has been focused on the case where the length of the generalized line(s) and the number of memory modules are equal. Motivation for this was presented at the beginning of Chapter 2 by considering computers designed as in Figure 3. When dealing with $[x,y]_N$-lines, setting the length of the generalized lines equal to the number of memory modules is an honest reflection of the real situation. In actual computations involving $[x,y]_N$-lines, and in particular the $[1,0]_N$-line and the $[0,1]_N$-line, what is often wanted is an entire row or column of a matrix. However, the number of processors in the actual computer dictates that the row or column can be processed no faster than N elements at a time, where the number of processors is taken to be equal to the number of memory modules, N. Since N elements of the row or column is all that can be fetched and

processed in one memory cycle, clearly $[x,y]_N$-lines are the natural candidates for consideration.

On the other hand, when a computation uses the generalized line L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2)), whose geometric realization is  -shaped, if the number of processors and the number of memories is greater than the length of the generalized line, then some components of the computer may just have to idle; the algorithm may not be able to utilize the full potential of the machine. Having extra hardware available, which cannot be utilized by an algorithm, may still result in improved performance, however. To see this, again consider an example presented in Chapter 2. Figure 12 indicated why the  cannot tesselate the plane, and hence there exists no valid skewing scheme, using only five memory modules, for the generalized line, L = ((0,0),(0,1),(0,2),(1,1),(2,1)). However, if the computer has more than five memory modules, perhaps a valid skewing scheme can be found which uses six or more memory modules. This motivates

Definition 12: A generalized line, L', is called a <u>cover</u> for another generalized line, L, if every component of L is contained in L'.

Notice that if $\varphi$ is valid for a cover of the generalized line, L', then $\varphi$ is also valid for L. The length of the generalized line used as a cover may conveniently be chosen to be the number of memory modules of the computer and then Theorem 5 can be applied to determine valid skewing schemes for the cover.

In general, given a collection of generalized lines, $\{L_1, L_2, \ldots, L_p\}$, not necessarily all of the same length, determination of the minimum number of memory modules, N, so that a valid skewing scheme, using N memory modules, exists for the collection, is quite difficult. A simple heuristic is to find a collection of generalized lines, $\{L_1', L_2', \ldots, L_p'\}$, such that $L_i'$ is a cover of $L_i$, the $L_i'$ are all of the same length, and such that they satisfy the conditions of Theorem 6. The resulting skewing scheme will also be valid for $\{L_1, L_2, \ldots, L_p\}$. There is often great latitude in choosing the covers, since given a generalized line, L, there may be many choices for L', so that L' tesselates the plane. For the generalized line, L = ((0,0), (0,1),(0,2),(1,1),(2,1)), if the search for covers that tesselate the plane is arbitrarily restricted to be polyominoes, and covers symmetric to other covers are ignored, there are still three generalized lines that tesselate the plane and are covers for L (see Figure 30). The rapid growth in the number of covers which are able to tesselate the plane indicates the formal evaluation of the heuristic given earlier may be very difficult.

> Conjecture: Given a polyomino, L, of length N, for which there is no valid skewing scheme using N memory modules, then there is a valid skewing scheme for L using N+1 memory modules if and only if there is a cover, L', for L, of length N+1, L' a polyomino, which tesselates the plane.

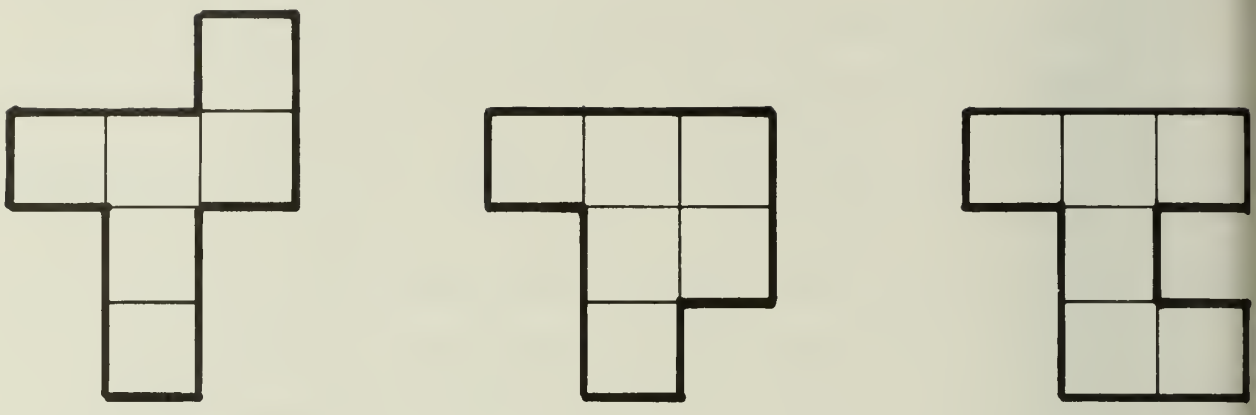


Figure 30: Covers for the Generalized Line ((0,0),(0,1),(0,2),(1,1),(2,1)) which Tesselate the Plane.

## 4.3 Comments on Broader Problems

In closing, it is useful to relate this research to the construction of real machines and to the work of others. Computers similar to the one modeled in Figure 3 have been built or proposed. Many researchers have realized that a very important problem is the construction of memory-processor connection networks [10,14]. In actual computations it is necessary to align the data so that the first component of an instance always appears at processor $i_1$, the second component of the instance always appears at processor $i_2$, etc. An example may be helpful. Consider the generalized line L = ((0,0),(0,1),(0,2),(1,1),(2,0),(2,1),(2,2)) and the periodic skewing scheme depicted in Figure 13. If an instance is fetched, whose designated element is stored in memory zero, then the data is already aligned, i.e. the element demanded by processor zero is in memory module zero, the element demanded by processor one is in memory module one, etc. For an instance whose designated element is stored in memory module one, however, the data from memory module one needs to be routed to processor zero, the data from memory module two needs to be routed to processor one, the data in memory module four needs to be routed to processor two, etc. This situation can be described as follows: If an instance of L is fetched, whose designated element is stored in memory module one, then to align the data with the processors, the memory-processor-connection network must be able to sort the permutation (1 2 4 0 5 6 3). The phrase "sort the permutation" is appropriate since paths must be established from processor zero to memory module one, from processor one to memory module two, etc., and this can be thought of as placing (1 2 4 0 5 6 3) as input at the processor side of the network and getting as output (0 1 2 3 4 5 6) at the memory side

of the network. Using this terminology, to be able to align any instance of L, using the skewing scheme of Figure 13, the network must be able to sort (0 1 2 3 4 5 6), (1 2 4 0 5 6 3), (2 4 5 1 6 3 0), (3 0 1 6 2 4 5), (4 5 6 2 3 0 1), (5 6 3 4 0 1 2), and (6 3 0 5 1 2 4). In general, if a periodic skewing scheme is used, derived from a tesselation generated by replicating parallelograms constructed as described in Section 4.1, then, in order to align the data, the memory-processor connection network must be able to sort N permutations. Standard fan-in arguments show that this will take $O(\log_2 N)$ time.

The results of Chapter 3 and Section 4.1 are very encouraging. It appears that in the most important practical cases valid linear skewing schemes exist if any valid skewing schemes exist. When linear skewing schemes are employed the memory-processor connection network is simpler. If it is unnecessary for the first component of the generalized line to go to processor zero, but if, on the other hand, it is sufficient that it always go to processor $i_1$, and similarly for the other components of the generalized line, then a network capable of performing arbitrary shifts is adequate. That is, the permutations to be sorted are just (0 1 2 ... N-1), (1 2 3 ... N-1 0), (2 3 4 ... N-1 0 1), ..., (N-1 0 1 ... N-2).

If only one generalized line is required for a computation, then there are no difficulties created by always sending the first component to processor $i_1$, the second component to processor $i_2$, etc. If several generalized lines are used by an algorithm, however, problems may develop. It is often the case, as in matrix multiplication, that the algorithm will require that the $j^{th}$ component of all the generalized lines used be sent to the same processor, $i_j$. A simple shifting network may no longer be

adequate, since the $j^{th}$ component of one generalized line may not be sent to the same processor as the $j^{th}$ component of some other generalized line. It is necessary to apply a corrective transformation after (or before) the shifting is performed. A different transformation may be required for each generalized line. For N, a power of two, Lawrie's $\Omega$-network [10] performs the shifting and the additional transformations needed simultaneously, without additional time delay or extra hardware. Unfortunately, as the results presented here indicate, when only the problem of finding valid skewing schemes is considered, a power of two is not the best choice for N. Taking N to be a prime number gives consistently better results. Taking N to be prime has two major disadvantages, however. The modular arithmetic is much slower and no networks that can perform as well as the $\Omega$-network are known.

One possible way of constructing such a memory-processor connection network is to follow the shifting network by a Benes̆ network [2] which can sort any permutation. The best algorithm for setting up the Benes̆ network is due to Opferman and Tsao-Wu [12]. This algorithm, which takes $O(N \log_2 N)$ time, is too slow for use on each memory cycle. However, since the generalized lines used by a program are fixed at compile time the way in which the network needs to be set up for each generalized line used by the program can be calculated once, before the program is run, stored in a memory, and then read out, to set the network up rapidly, when needed. Adding a Benes̆ network to the memory-processor connection network, as just described, will create additional cost and will slow down the machine somewhat since the data will have to pass through more gates. This problem, and particularly incorporating the Benes̆ network in with the shifting network, is a good candidate for further research.

# LIST OF REFERENCES


[1] Batcher, K. E., "The Multi-dimensional Access Memory in STARAN," Presented at the 1975 Sagamore Conference on Parallel Processing and submitted for publication in the IEEE Transactions on Computers Special Issue on Parallel Processing.

[2] Beneš, V. E., Mathematical Theory of Connecting Networks and Telephone Traffic, Academic Press, New York; 1965.

[3] Budnik, P. and D. J. Kuck, "The Organization and Use of Parallel Memories," IEEE Transactions on Computers, Vol. C-20, No. 12, pp. 1566-1569; December 1971.

[4] Chandra, A. K., "Independent Permutations, as Related to a Problem of Moser and a Theorem of Pólya," Journal of Combinatorial Theory, Series A, Vol. 16, pp. 111-120; 1974.

[5] Gardner, M., "Mathematical Games; More About Tiling the Plane: The Possibilities of Polyominoes, Polyiamonds and Polyhexes," Scientific American, Vol. 233, No. 2, pp. 112-115; August 1975.

[6] Golomb, S. W., Polyominoes, Charles Scribner's Sons, New York; 1965.

[7] Hardy, G. H. and E. M. Wright, An Introduction to the Theory of Numbers, Oxford University Press, London; 1954.

[8] Hoffman, K. and R. Kunze, Linear Algebra, Prentice-Hall, Inc., Englewood Cliffs, New Jersey; 1961.

[9] König, D., Theorie der Endlichen und Unendlichen Graphen, Akademische Verlagsgesellschaft M.B.H., Leipzig; 1936.

[10] Lawrie, D. H., "Memory-Processor Connection Networks," Ph.D. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-73-557; February 1973.

[11] Muraoka, Y., "Storage Allocation Algorithms in the TRANQUIL Compiler," M.S. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. 297; January 1969.

[12] Opferman, D. C. and N. T. Tsao-Wu, "On a Class of Rearrangeable Switching Networks," Bell System Technical Journal, Vol. 50, pp. 1579-1618; May-June 1971.

[13] Pólya, G., "Über die 'doppelt-periodischen' Lösungen des $n$-Damen-Problems," in W. Ahrens, Mathamatische Unterhaltungen und Spiele, Teubner, Leipzig, pp. 364-374; 1918.

[14] Swanson, R. C., "Interconnections for Parallel Memories to Unscramble $p$-Ordered Vectors," IEEE Transactions on Computers, Vol. C-23, No. 11, pp. 1105-1115; November 1974.

[15] Thornton, J. E., Design of a Computer the Control Data 6600, Scott, Foresman and Company, Glenview, Illinois; 1970.

[16] Yao, F., Private Communication.

# VITA

Henry David Shapiro was born in New York, New York on September 17, 1947. He received his bachelor of arts degree in mathematics from the Johns Hopkins University, graduating with both honors and departmental honors. At that time, he was also elected to Phi Beta Kappa. In October 1969, Mr. Shapiro was awarded a master of science degree in mathematics from Stanford University. Before coming to the University of Illinois, he taught high school mathematics and computer science at West Springfield High School, Fairfax County, Virginia, from September 1969 until June 1972. While working on his doctorate, he held a National Science Foundation Graduate Fellowship and a Research Assistantship in the Department of Computer Science. Mr. Shapiro is a member of Sigma Xi, the Association for Computing Machinery, and the National Council of Teachers of Mathematics. He has had two papers accepted for publication: "Storage Schemes in Parallel Memories" (to appear in the Proceedings of the 1975 Sagamore Computer Conference on Parallel Processing) and "A New Approach to Teaching a First Course in Compiler Construction" with M. D. Mickunas (to appear in the Proceedings of the Sixth Symposium on Computer Science Education).

| BIBLIOGRAPHIC DATA SHEET | 1. Report No. UIUCDCS-R-75-776 | 2. | 3. Recipient's Accession No. |
|---|---|---|---|
| 4. Title and Subtitle: THEORETICAL LIMITATIONS ON THE USE OF PARALLEL MEMORIES | | | 5. Report Date: December, 1975 |
| | | | 6. |
| 7. Author(s): Henry David Shapiro | | | 8. Performing Organization Rept. No. |
| 9. Performing Organization Name and Address: Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois | | | 10. Project/Task/Work Unit No. |
| | | | 11. Contract/Grant No.: NSF GJ 41538 |
| 12. Sponsoring Organization Name and Address | | | 13. Type of Report & Period Covered: Ph.D. Thesis |
| | | | 14. |

15. Supplementary Notes

16. Abstracts

Regardless of the underlying machine architecture, independently addressable memory modules contribute significantly to program speed-ups on modern computers. Because of memory conflicts which arise while accessing data, actual program speed-ups are generally less than theoretically possible. Organizing the data of a computation so as to avoid memory conflicts is particularly difficult for data which can logically be viewed as two-dimensional. Several geometric and algebraic conditions are presented which determine if the data of a computation can be organized to avoid memory conflicts. It is shown that a prime number of memory modules gives higher memory utilization and allows the use of simpler storage schemes than a power of two number of memory modules. The case of greatest practical significance, references to rows, columns and diagonals of a matrix, is given special attention. Finally, a brief discussion is presented which relates this research to that of a companion problem, the construction of memory-processor connection networks for single-instruction-multiple-data stream machines.

17. Key Words and Document Analysis. 17a. Descriptors

parallel memories
memory conflicts
skewing schemes
linear skewing
tesselations

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

| 18. Availability Statement | 19. Security Class (This Report) UNCLASSIFIED | 21. No. of Pages |
|---|---|---|
| | 20. Security Class (This Page) UNCLASSIFIED | 22. Price |

FEB 23

BOUND BY
HNM

UNIVERSITY OF ILLINOIS-URBANA
510.84 IL6R no. C002 no.776-781(1975
Theoretical limitations on the use of pa
3 0112 088402505